3D scene analysis through non-visual cues

Aron Monszpart

An upgrade report submitted in partial fulfillment

of the requirements for the degree of

Doctor of Philosophy

of

University College London.

Department of Computer Science

University College London

Oct 6, 2019

I, Aron Monszpart, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the work.

Abstract

The wide applicability of scene analysis from as few viewpoints as possible attracts the attention of many scientific fields, ranging from augmented reality to autonomous driving and robotics. When approaching 3D problems in the wild, one has to admit, that the problems to solve are particularly challenging due to a monocular setup being severely under-constrained. One has to design algorithmic solutions that resourcefully take advantage of abundant prior knowledge, much alike the way human reasoning is performed. I propose the utilization of non-visual cues to interpret visual data. I investigate, how making non-restrictive assumptions about the scene, such as "obeys Newtonian physics" or "is made by or for humans" greatly improves the quality of information retrievable from the same type of data. I successfully reason about the hidden constraints that shaped the acquired scene to come up with abstractions that represent likely estimates about the unobservable or difficult to acquire parts of scenes. I hypothesize, that jointly reasoning about these hidden processes and the observed scene allows for more accurate inference and lays the way for prediction through understanding. Applications of the retrieved information range from image and video editing (*e.g.,* visual effects) through robotic navigation to assisted living.

Acknowledgements

First and foremost, I would like to express my humble gratitude to the primary advisor of this PhD, Prof. Niloy J. Mitra. There are very few people in the world who taught me more than he did. It is an honor to have got the opportunity to work with him and I am especially grateful for the amount of time he spent on helping me without reservations and consideration for time of day, week or year. Thank you Niloy!

I also thank the secondary advisor of this PhD, Prof. Gabriel Brostow for always, when asked, providing his academically very refreshing viewpoint and critical scrutiny allowing me to think outside the box.

Thanks to all my collaborators, I am truly grateful for the time they spent helping and advising me. I'd like to thank all reviewers who helped improve the works presented in this thesis for their constructive comments and suggestions. Thanks to UCL, especially its support staff, Dawn Bailey, Wendy Richards and JJ Giwa-Majekodunmi for allowing me to only focus on the research. I of course thank all my fellow PhD students and friends who have provided invaluable discussions and opportunities to learn during this critical period.

Special thanks to the SmartGeometry group: Melinos Averkiou, Paul Guerrero, James Hennessey, Moos Hueting, Carlo Innamorati, Tom Kelly, Bongjin Koo, Dan Koschier, Martin Kilian, Nicolas Mellado, Robin Roussel, Tuanfeng Y. Wang, and Zachary Bodnár, Gabriel Brostow, Neill Campbell, Nathan Carr, Jose Echevarria, Katalin Farkas, Michael Firman, Marios Fournarakis, Sebastian Friston, Stephan Garbin, Clément Godard, Peter Hedman, Corneliu Ilisescu, Geoffrey Jones, Simon Julier, Byungmoon Kim, Vladimir Kim, Florent Lafarge, Hui Lin, Oisin Mac Aodha, Drew MacQuarrie, Dushyant Mehta, Grégory Rogez, Chris Russell, Manolis Savva, Neil Smith, Julian Straub, Zsófia K. Takács, Denis Tomè, Daniyar Turmukhambetov,

David Vanderhaghe, Lucy Vanes, Jamie Watson, and Rui Yu for their invaluable help, comments, support and discussions.

I also thank the support of the funding bodies: ERC Starting Grant SmartGeometry (StG-2013-335373), ERC PoC Grant (SemanticCity), ERC Starting Grant realFlow (StG-2015-637014), Marie Curie CIG, ANR Mapstyle project (ANR-12-COORD-0025), EU project CR-PLAY (no 611089) `www.cr-play.eu`, NSFC (No. 61402402), Royal Society Advanced Newton Fellowship, UCL Impact, Google Faculty Awards, a Google PhD Fellowship, and gifts from Adobe. Thanks Moos for being there to teach me about what's worth living for. Thanks Zack for teaching me the value of communication and time. Thanks to Moos and Nicolas for inspiring me to becoming a better engineer.

I will never forget to be grateful to my teachers and mentors, without whom I would not have gotten this far: Dr. Ferenc Vajda, Gábor Horváth, Miklós Hóbor, Judit Papp, Dr. Borbála Pintér, and Sarolta Tóth.

None of this would have been possible without my inexhaustibly generous and patient family: my parents who did everything they could to let and make me chase my dreams, my brother, who was always there to help, and Veronika, my eternal love, thanks for withstanding this long period and always being there to cheer me up with your smile.

Impact statement

In this thesis, I present four novel approaches to 3D scene analysis utilizing non-visual cues. Each approach was presented at SIGGRAPH or SIGGRAPH Asia, the premier venues for scholarly work in the field of computer graphics. To encourage future academic work, results and datasets have been released for the methods in Chapters 3 to 6.

Publications

This thesis encompasses the following publications:

Chapter 3. MONSZPART, A., MELLADO, N., BROSTOW, G., AND MITRA, N. J. RAPter: Rebuilding Man-made Scenes with Regular Arrangements of Planes. *ACM SIGGRAPH 34*, 4 (July 2015), 103:1–103:12

Chapter 4. SHAO*, T., MONSZPART*, A., ZHENG, Y., KOO, B., XU, W., ZHOU, K., AND MITRA, N. J. Imagining the Unseen: Stability-based Cuboid Arrangements for Scene Understanding. *ACM SIGGRAPH Asia 33*, 6 (Nov. 2014), 209:1–209:11. * Joint first authors

Chapter 5. MONSZPART, A., THUEREY, N., AND MITRA, N. J. SMASH: Physics-guided Reconstruction of Collisions from Videos. *ACM SIGGRAPH Asia 35*, 6 (Nov. 2016), 199:1–199:14

Chapter 6. MONSZPART, A., GUERRERO, P., CEYLAN, D., YUMER, E., AND MITRA, N. J. iMapper: Interaction-guided Scene Mapping from Monocular Videos. *ACM SIGGRAPH 38*, 4 (July 2019), 92:1–92:15

Contents

1 Introduction **12**

1.1 Overview 17

1.2 List of Contributions 21

2 Related work **23**

2.1 Analysis of static scenes 23

2.2 Analysis of dynamic scenes 30

3 Regularity in raw scans as non-visual cue **35**

3.1 Introduction 36

3.2 Overview 38

3.3 Algorithm 40

3.3.1 Initialization 40

3.3.2 Candidate generation 40

3.3.3 RAP selection 42

3.3.4 Irregularity revisited 44

3.4 Implementation Details 46

3.5 Evaluation 51

3.5.1 Comparisons 51

3.5.2 Results 53

3.5.3 Quantitative evaluation 59

3.6 Conclusions 61

4 Physical stability as non-visual cue **63**

4.1 Introduction 63

4.2 Overview . . . 65
4.3 Algorithm . . . 66
4.3.1 Generating Initial Cuboids . . . 67
4.3.2 Inferring Inter-cuboid Contact Types . . . 70
4.3.3 Refining Cuboid Parameters under Contact Constraints . . . 75
4.4 Evaluation and Results . . . 76
4.4.1 Synthetic Data . . . 77
4.4.2 Performance on NYU2 Dataset . . . 79
4.4.3 Cluttered Indoor Scenes . . . 81
4.4.4 Multi-view Recordings . . . 82
4.4.5 Comparison . . . 83
4.4.6 Performance . . . 84
4.5 Applications . . . 85
4.6 Conclusion . . . 86

5 Accurate dynamics as non-visual cue 88
5.1 Introduction . . . 88
5.2 Formulation . . . 90
5.2.1 Rigid Bodies . . . 90
5.2.2 Problem Statement . . . 92
5.2.3 Center of Mass Trajectories . . . 93
5.2.4 Orientation and Angular Momentum . . . 95
5.2.5 Energy Limiting . . . 96
5.2.6 Conservation of Momentum . . . 96
5.3 Method . . . 98
5.3.1 Parameter Estimation . . . 98
5.3.2 Initializing Positions . . . 103
5.3.3 Initializing Orientations . . . 107
5.3.4 Implementation . . . 108
5.4 Evaluation . . . 109
5.4.1 Validation on Synthetic Data . . . 109
5.4.2 Testing on Real Sequences . . . 112

5.4.3 Real-world Cases . . . 113
5.4.4 Staging collisions . . . 117
5.4.5 Limitations and Discussion . . . 118
5.5 Discussion . . . 119

6 Human interactions as non-visual cue 124
6.1 Introduction . . . 124
6.2 Scenelets: Representation and Descriptors . . . 127
6.2.1 Scenelet Representation . . . 127
6.2.2 Scenelet Descriptors . . . 128
6.2.3 Interaction-saliency for Scenelets . . . 129
6.3 Algorithm Overview . . . 130
6.4 Identifying Informative Interactions . . . 132
6.4.1 Generating an Initial Skeletal Estimate . . . 132
6.4.2 Estimating Interaction-likelihood Score . . . 133
6.5 Retrieving matched interactions . . . 134
6.6 Scene Mapping via Global Optimization . . . 135
6.6.1 Video Segments with High Interaction-likelihood . . . 136
6.6.2 Video Segments with Low Interaction-likelihood . . . 137
6.6.3 Scene Mapping via a Global Optimization . . . 138
6.6.4 Multi-person scenes . . . 141
6.7 Results and Discussion . . . 142
6.7.1 Occluded Interaction Benchmark Dataset (**i**3DB) . . . 145
6.7.2 Evaluating Object Placement Quality . . . 147
6.7.3 Evaluating Actor Pose Quality . . . 148
6.7.4 User Study . . . 152
6.7.5 Ablation Study . . . 152
6.7.6 Limitations . . . 153
6.8 Conclusion and Future Work . . . 155
6.8.1 Future Directions . . . 156

7 Discussion 158
7.1 Regularity in raw scans . . . 158

7.2 Physical stability 160
7.3 Accurate dynamics 160
7.4 Human interactions 162
7.5 Non-visual cues 165

Bibliography **165**

Chapter 1

Introduction

Our desire for tools to render our lives safer and less laborious can be considered as old as our consciousness. In the current age of automation it has become a realistic expectation of machines not only to follow our instructions in deterministic environments but also to autonomously operate in dynamic environments with their goals and the measure of their success being specified by instructions with an ever decreasing level of specificity and increasing level of abstraction and symbolism — characteristics of communication otherwise so natural to us humans.

> "*If every instrument could accomplish its own work, obeying or anticipating the will of others, like the statues of Daedalus, or the tripods of Hephaestus, which, says the poet, "of their own accord entered the assembly of the Gods;" if, in like manner, the shuttle would weave and the plectrum touch the lyre without a hand to guide them, chief workmen would not want servants, nor masters slaves.*" — *Aristotle: Politics I:IV*

We expect such autonomous systems to be able to observe their surroundings and maintain matching internal models possessing sufficient representation power for the purposes specified by the aforementioned measures of success. Additionally, a powerful system will not only be able to parse the current state of the world, but also will be able to speculate about future states of their environment.

An indispensable requirement of the internal models these systems maintain is the capability of representing relations among the models' building blocks (*e.g.,* objects or actors), *i.e., understanding* the environments up to the desired level of complexity. More specifically, it is not only important to acquire an accurate enough representation

of the parts of the environment that can be and are being observed, but to analyze why the environment arrived to its current state and what constraints manifest themselves in space and time between its building blocks influencing the environment's future evolution.

For the purposes of this discussion, I consider systems designed and used by humans to perform tasks that represent utility to humans. I do not include design considerations that would arise when optimizing for machine-to-machine communication or systems benefiting non-human natural entities. I denote a specific instance of the environment that is being observed and modeled a *scene* and the act of estimating the model parameters *analysis*. An *object* is a self-contained building block of a scene that often holds semantic utility to humans describing goals for autonomous systems. I also restrict the notion of observing a scene to only include active or passive light based acquisition methods that can be applied utilizing commodity, customer-grade hardware (laser scanners, RGBD sensors and smartphone video cameras). I define *direct* observation as the act of operating such capture devices without modification as they were designed to be used. The *directly* observable parts of scenes refer to the information capturable by such acquisition devices as an outcome of at most a few capture sessions from one or more, physically easily accessible viewpoints.

The reductive disambiguations of the above terms are put in place to ensure our technological convergence towards the goal of equipping autonomous robots with tools that are considerate of their often present energy budget constraints and mobility restrictions ensuring risk-mitigation of catastrophic failures in unknown environments.

In this thesis I show multiple examples of scene analysis models, where extending the constructs with representations of specific domains and types of constraints and identifying the instances of such constraints in the specific scenes, including estimating their relevant parameters, yields great benefit both for the completeness of the autonomous systems' abstract images of the scenes as well as for predicting the future evolution of scenes with reduced levels of uncertainty. The most emphasized virtue of the designed models to follow is their capability to hypothesize about the not *directly* observable parts of the targeted scenes via novel applications of non-visual cues. The estimated model parameters extend to regions of the scenes that are difficult to observe due to spatial restrictions as well as to modalities of information that cannot be captured by light

based acquisition devices and thus parameters in the domains of these newly included modalities of information can only be inferred. The modalities of information that most often occur in this thesis include relations between scene objects, such as perfect relative orientation in the context of spatial alignment or physically supporting in the context of statics, internal physical parameters characteristic to objects in the scenes, and consistent interactions between humans and their surrounding objects.

Hypothesis. The main claim of this thesis stipulates that reasoning about the scenes through the aforementioned, extended collection of modalities covering not directly observable domains of information not only expands our knowledge of the observed scenes with additional information in domains previously rarely considered in combination with 3D scene analysis, but also improves the coverage and accuracy of the resulting estimates of the directly observable parts of the scenes. The hypothesized reason for the improvement is that the additional constraints added to the models manifest in new, spatially and or temporally non-local relations between actors in the scenes that equip the parameter estimation optimizations with the capability to ensure wider adherence to prior knowledge derived from non-visual cues and better coherence among the parts of the scenes.

Requirements of success. A system, that is more successful than any existing baseline method at analyzing scenes in 3D based on data captured from at most a few viewpoints should fulfill a number of basic requirements. It should be able to model the directly observable regions of a scene at least as accurately as any existing baseline method, it has to clearly define the additional modality of information to which it extends its reasoning, it should define a model that is capable of efficiently and concisely capturing the important aspects of the additional information, it should verifiably estimate consistent or even accurate parameterizations of the directly not observable parts of the scenes, and the quality of its outputs should degrade gracefully when encountering increasing amounts of uncertainty.

Challenges. The most important challenges when aiming to acquire real environments with high accuracy in 3D is sensor noise, occlusion and ambiguity due to partial observations. The effect of the first might be mitigated by employing multiple types of sensors, multiple capture sessions or increasing the number of viewpoints. For example, one can get closer to the objects in a scene to align the sensors' spatial resolution focal

point with the missing details in question. The naïve solution of increasing the number of captures might also resolve a fraction of the second problem, occlusion. However, the strategy of increasing the number of viewpoints has many drawbacks: it might not be possible to access the scene in a non-disruptive way over the longer period of time required, the scene might be so rapidly changing that assumptions about temporal consistency will be violated, or the resulting method of acquisition simply will not respect the requirements for energy efficiency.

In real life scenarios the scenes of interest are often so cluttered and complicated that large fractions simply cannot be fully observed because they are not accessible given the size of the capture device or other limitations arising due to human factors, *e.g.,* close-ups could be considered invasive and even destructive with regard to the observed environment in question. It is therefore an important objective of the work conducted in this thesis to address the problems arising from occlusion by explicitly modeling the not directly observable parts of the scenes.

Ambiguity is another important challenge to address when optimizing for as few viewpoints as possible in order to maximize the utility of a designed system. The challenge of ambiguity arises due to the fact that our observations of the scene are partial, the entirety of the scene has not been observed. The unobserved sections of the scene may thus have multiple explanations, all allowed for by the observations that were possible to gather, *i.e.,* the space of valid solutions is under-constrained. A prevailing approach to resolve such ambiguity is to carefully choose widely-applicable, non-restrictive pieces of prior cues and expand the designed models to include the parameterizations derived from such pieces of prior knowledge. The discussed models in this thesis clearly describe the additional assumptions that grant the methods the capability to resolve some of the ambiguity in the individual problem settings. Each of the following chapters presents a case, where surprisingly simple pieces of information or laws from well-established research areas of physics allow the design of methods that are capable of resolving ambiguity with previously unseen power.

Figure 1.1 shows an example of a scene from a real scenario that well represents the above mentioned challenges. We expect a 3D scene analysis algorithm, when given the information that parallel and right-angle relations represent special value to us, to be able to infer that there is a main coordinate frame in the scene shared by the floor,

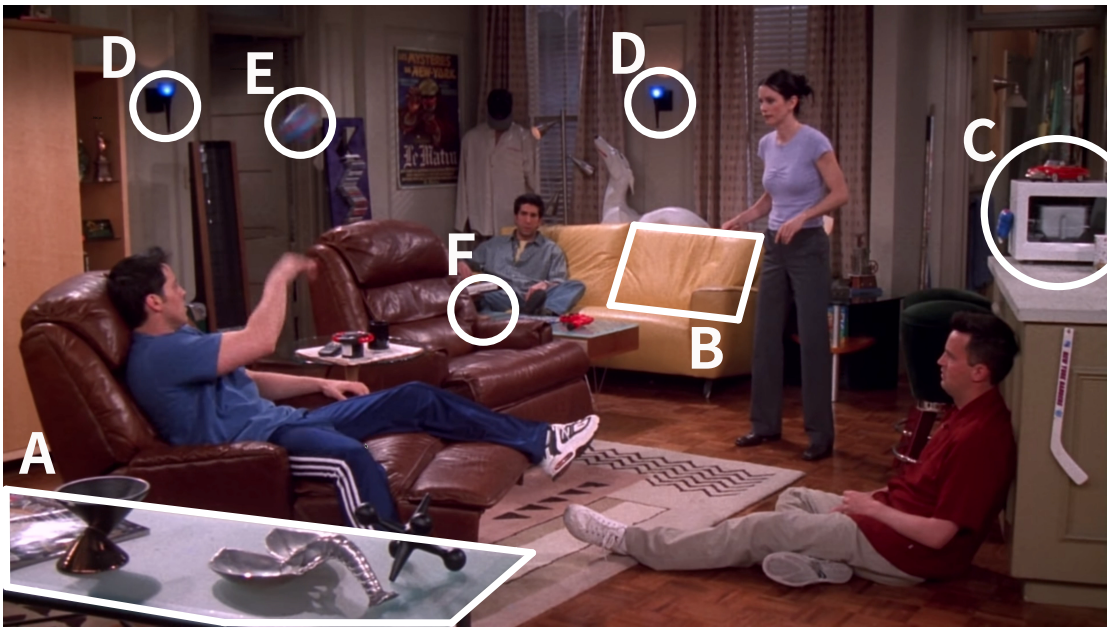


Figure 1.1: Challenges arising due to occlusion in scene analysis. (A) We expect the plane of the nearby table-top to be exactly parallel to the plane of the floor when the room is scanned with a depth sensor. (B) Even when relatively sparsely sampled, the couch back should retain its orientation and not be regularized to one of the main wall or floor orientations in the scene. (C) There should exist a physically stable explanation of the scene, where the microwave is not 'glued' to the wall but is supported by the kitchen bench standing on the floor. (D) Wall lamps should be assumed to be 'fixed' to the walls. (E) The 3D trajectory of the ball thrown observed in a monocular video from a fixed viewpoint should be physically valid, obeying Newtonian dynamics. (F) The occluded knee and ankle joints of the person should not intersect any of the furniture in a hypothesized scene layout. Image source: Friends©.

ceiling, and the walls of the room. We also expect that the 3D representation of a few pieces of furniture, such as the glass table in the front or the kitchen counter on the right should be in a perfectly "parallel" relationship with the aforementioned floor and ceiling orientations. Simultaneously, other objects, such as the sofas and statues should not be aligned to the main orientations in the scenes and their shape and orientation should be analyzed and processed by methods outside the scope of this thesis.

We also expect a parameterized model of such a scene to obey the laws of physics. Firstly, in terms of statics representations of objects that are not moving should be directly or indirectly supported by the floor, the walls, or the ceiling. When attempting to come up with an accurate explanation in the form of a network of support relationships, one often needs to consider the objects in their non-local context. *E.g.*, the microwave on the right can only be supported by the kitchen counter if the kitchen counter is also supported by the floor. Similarly, the mug on the glass table between the two brown

sofas should be supported by the glass table. This glass table with the occluded leg either has to stand on an occluded object or its leg has to extend down to the floor.

Secondly, in terms of dynamics we expect the flying ball to obey the relevant conservation laws and have a perfect parabola trajectory. Even if the motion blur in Figure 1.1 (E) can be significantly reduced by using capture devices as common as smartphones, without adding the knowledge of the gravitational acceleration constant on Earth, an infinite number of 3D parabolas exist that would reproject to the 2D track computable from the ball projections.

Lastly, we expect the motion of the humans to be realistic, *i.e.,* not intersect the objects in the scene in 3D even around strong occlusion Figure 1.1 (F). Furthermore, where relevant the 3D motion of the humans in the scene should to resemble the typical ways people interact with the types of objects in the scene. It's worth noting that in the scene, where this specific footage was captured, no new acquisition sessions can be initiated anymore. Information for all new purposes, such as digital renovation, conversion to 3D cinematography, or physical animations in derived games have to be inferred from the existing, monocular video footage.

1.1 Overview

A common trait shared by the methods to be presented is that they model spatially, temporally or spatiotemporally non-local relations between different parts of the scenes.

With the proliferation of acquisition devices, gathering massive volumes of 3D data has become increasingly easy. Processing such large masses of pointclouds, however, remains a challenge. Large pointclouds are particularly a problem for raw 3D scans with missing data, noise, and varying sampling density. In Chapter 3, we present a simple, scalable, yet powerful data reconstruction algorithm as a first example of utilizing cues that are not directly observable. The work focuses on reconstruction of man-made scenes as regular arrangements of planes (RAP), thereby selecting both local plane-based abstractions along with their non-local inter-primitive relations. The work proposes a novel selection formulation to directly balance between data fitting and the simplicity of the resulting arrangement of extracted planes. The main technical contribution is a formulation that allows less dominant orientations to still retain their internal regularity and not become overwhelmed and regularized by oversampled, dominant

scene orientations. The approach is evaluated on a variety of complex 2D and 3D pointclouds demonstrating the advantages over previous methods. Our system takes a list of preferred, pairwise relations, a raw 3D pointcloud, a regularization weight, and a minimum spatial feature scale as input. The preferred relations are defined as a function of two planar primitives in 2D or 3D. The function is required to return a continuous scalar measure of how imperfect the relation is between the two primitives, and is expected to be zero if and only if the relation between two primitives is perfect. We show examples, where the input perfect relations are parallelity, perpendicularity and have relative angles being multiples of $\pi/3$. The output of the method is a collection of primitives, *i.e.,* lines or planes with no extent, the identified regular relations and the association of points in the input pointcloud to the fit primitives. Regularity, *i.e.,* the assignment of special importance to certain pairwise relations is the non-visual cue that is added to the modeling in a novel manner in Chapter 3. The work was published as [107].

Missing data due to occlusion is a key challenge in 3D scene analysis, particularly in cluttered man-made scenes. Such partial information about the scenes limits our ability to analyze and understand them. The work presented in Chapter 4 abstracts such environments as collections of cuboids and hallucinates geometry in the occluded regions by non-locally analyzing the physical stability of the resultant arrangements of the cuboids. The algorithm extrapolates the cuboids into the unseen regions to infer both their corresponding geometric attributes (*i.e.,* size and orientation) and how the cuboids topologically interact with each other (*i.e.,* are touching or fixed to each other). The resultant arrangement provides an abstraction for the underlying structure of the scene that can then be used for a range of common 3D scene analysis tasks. The algorithm is evaluated on a large number of test scenes with varying complexity, results are validated on existing benchmark datasets, and we demonstrate the usefulness of the recovered cuboid-based structures with respect to object retrieval, scene completion, and scene editing tasks. In addition to an RGBD image the method takes as input the cuboid primitives in the scene that have infinite mass, *i.e.,* the floor, ceiling and walls. Optionally, the user may also provide 2D strokes as annotations of the input image, which are then used to construct the initial cuboids of the scene. The output of the method is a physically stable collection of cuboids including the types of relations.

Physical stability represents the non-local, non-visual cue added to the modeling process in Chapter 4. The work shows how one has to extend modeling beyond pairwise relations in order to achieve enough representational power to be able to model realistic indoor environments. The work was published as [146].

Collision sequences are commonly used in games and entertainment to add drama and excitement. Staging even two body collisions in the real world can be difficult, as one has to get timing and the object trajectories to be correctly synchronized. After tedious trial-and-error iterations, when objects can actually be made to collide, they remain being difficult to capture in 3D. In contrast, synthetically generating realistic collisions is difficult as it requires adjusting different collision parameters (*e.g.,* object mass ratio or coefficients of restitution) and appropriate initial parameters. Chapter 5 presents SMASH to directly *read off* appropriate collision parameters from raw, monocular input video recordings. Technically, this is enabled by utilizing laws of rigid body collision to regularize the problem of lifting 2D trajectories to a physically valid 3D reconstruction of a collision. The reconstructed sequences can then be modified and combined to easily stage novel and consistent collisions. The system is evaluated on a range of synthetic scenes and the effectiveness of the method is demonstrated by accurately reconstructing several complex real world collision events. The key insight resides in the fact that the motion states of objects in free-fall before and after a collision together well describe the interaction itself. We make the simplifying assumption not to model air-drag in Chapter 5 and thus use "free-fall" as it is defined in Newtonian physics to refer to the projectile motion of objects having only gravity as sole force acting on them[1]. There is a spatio-temporally non-local relationship between the dynamic parameters of the objects before and after collision. Whilst it is often difficult to accurately observe an interaction between two objects due to the arising self-occlusion and naturally high speed of events, it is less difficult to collect multiple measurements of the motion of objects when they are well separated in space. An indispensable enabler of performing such estimations without specialized lab equipment is smartphones' recently arisen capability of capturing videos at high frame rates. The input to the system is

i) a high-framerate (240 FPS) RGB video of two free-form, 3D objects colliding with each other exactly once, recorded from a static viewpoint,

[1] `http://wikipedia.org/wiki/Free_fall`

ii) textured 3D scans of the 3D objects acquired separately, and

iii) at least two annotations of both objects' 3D orientation before and after collision (totaling eight annotations).

The orientation annotations are expected to be within a few degrees of error and we assume that the objects have undergone less than a full turn between the two annotations. The scans of the objects are only used to enable the annotation of the orientations and to approximately compute the inertia tensor of the objects. The output of the method is a physically accurate, analytic parameterization of the 6 DoF motion of both objects and an estimate of the mass ratio and coefficient of restitution between the two objects. In Chapter 5 dynamically accurate motion and internal physical properties are the non-visual cues employed in a novel manner. The work was published as [108], and further lead to contribution to [30] and [31].

Next generation smart and augmented reality systems demand a computational understanding of monocular footage that captures humans in physical spaces to reveal consistent object arrangements and human-object interactions. Despite recent advances, both in scene layout and human motion analysis, the above setting remains challenging to analyze due to regular occlusions that occur between objects and human motions. The key observation is that *interactions* between object arrangements and human actions is often strongly correlated and hence can be used to help recover from these occlusions. Chapter 6 presents iMAPPER, a data-driven method to identify such human-object interactions and utilize them to infer layouts of occluded objects. Starting from a monocular video with detected 2D human joint positions that are potentially noisy and occluded, iMAPPER first introduces the notion of *interaction-saliency* as space-time snapshots where informative human-object interactions happen. Then, the work proposes a non-local optimization to retrieve and fit interactions from a pre-computed database to the detected salient interactions in order to best explain the input video. The approach is extensively evaluated, both quantitatively against manually annotated ground truth and through a user study and we demonstrate that iMAPPER produces consistent scene layouts for scenes with medium to heavy occlusion. The input to the system at test time is

i) an RGB video of one or more actors interacting with their environment recorded from a single viewpoint and

ii) the output of a monocular 3D human pose estimation method that emits 2D joint detections, their detection confidence, and pelvis-centered 3D poses.

The input set of 3D poses may be sparse in time, *i.e.,* 3D poses may only exist for video frames with no occlusion, not all video frames with 2D detections are required to have a 3D pose associated with them. Additional input to the system *before* test time, subject to a pre-processing step is a collection of recordings of 3D skeletons interacting with their environment and annotations of the objects of that environment containing oriented 3D bounding boxes of the objects and their category labels. The output of the system are 3D skeleton poses for every frame with 2D detections and a consistent arrangement of oriented bounding boxes of 3D objects with category labels that represent a scene that is compatible with the human motion appearing in the video. The object categories and placements are not primed to match the visual appearance of the scene in the video as iMAPPER solely relies on human pose detections derivable from the input video. The non-visual cues employed in a novel manner in Chapter 6 are the typical ways of interaction between humans and the objects surrounding them. The work was published as [106].

1.2 List of Contributions

The specific, novel contributions presented in this thesis are:

i) The concept of representative, higher-order graph nodes, that allow for easier balancing between regularization and data-fidelity when abstracting large 3D pointclouds with planar primitives (Chapter 3).

ii) Stability as a non-local property and modeled on the scene level (see Figure 2.2). Doing so allows us to gain physical understanding of static 3D scenes from single-shot RGBD images (Chapter 4).

iii) Newtonian physics has analytical parameterizations of motion that can represent physically accurate collisions in 3D. Using such compact parameterizations, collisions can be realistically reconstructed from high framerate RGB videos (Chapter 5).

iv) Typical interactions between humans and objects in their environments are so distinct, that a few 2D joint detections over some time are enough to successfully lookup realistic interaction snippets from a pre-computed database of 3D human-object

interactions. Such snippets can then be combined to consistent 3D scenes and realistic 3D human motion.

The work presented in Chapter 3 was developed and written in a collaboration with multiple parties and published as [107]. Specifically, Nicolas Mellado developed the method for merging nearby, co-planar primitives and the visualizations of the output 3D pointclouds.

Similarly, the work presented in Chapter 4 was developed and written in a collaboration with multiple parties, and published as [146]. "Image-guided cuboid improvement of initial cuboids" in Section 4.3.1, Sections 4.3.2 and 4.3.3, and Section 4.5 i)-ii) were implemented by joint first author Tianjia Shao, Section 4.5 iii) was implemented by Youyi Zheng.

Furthermore, the work presented in Chapter 5 was developed and written in a collaboration with multiple parties, and published as [108]. Nils Thuerey provided the example implementation of the impulse collision model used to verify the correctness of the optimization in the case of synthetic outputs.

Finally, the work presented in Chapter 6 was developed and written in a collaboration with multiple parties, and published as [106]. Paul Guerrero designed and developed the motion, object, and scenelet descriptors described in Section 6.2. Paul also designed and implemented the differentiable loss function for the occlusion term defined in Equation (6.3), developed the baseline method presented as 'GT+MD' in Table 6.1, and designed and conducted the user study in Section 6.7.4. Duygu Ceylan developed the baseline action-recognition method re-trained from Luvizon et al. [97] and compared to in Table 6.3.

Chapter 2

Related work

In the following, I summarize, how other projects have attempted to use additional prior information, such as abstraction by geometric primitives or knowledge about the physical processes shaping the captured surroundings, to advance towards an in-depth understanding of the observed environment including its actors and their relations.

I cover existing solutions aiming to abstract large datasets of raw 3D pointclouds in order to compress and repair noisy captures before being passed to higher-level scene understanding methods, physical stability based 3D scene understanding methods as well as physics- and humans-oriented analysis of dynamic scenes.

2.1 Analysis of static scenes

Analysis of raw pointclouds. With the growth of easy to use acquisition devices (*e.g.,* Kinect®) scene understanding involving object segmentation and labeling of indoor scenes has received much attention in recent years. The size and volume of recorded pointclouds continue to evolve rapidly ([117, 116]). In order to distill such vast amounts of raw data to useful knowledge, researchers have looked beyond surface reconstruction towards data abstraction and analysis. Various solutions have been proposed, both in supervised and unsupervised settings [4, 84, 150].

Urban reconstruction. Many man-made scenes tend to explicitly conform to the axes of one or a few coordinate frames by design. Buildings and rooms typically come with such canonical reference frames. This observation is often exploited in the 'Manhattan assumption', wherein planes are fitted, but are restricted to particular frame directions. Such regularities manifest as redundancy in data, which can be exploited to denoise and consolidate measurements. Based on this observation, Gallup et al. [43]

presented a multi-view plane-sweep-based stereo algorithm to recover planar surfaces, potentially slanted, using a GPU-assisted real-time approach. They rely on finding a single ground plane, projecting points from upright objects onto that plane, and then finding the orientation that minimizes the entropy of those points onto an L-frame. The 'Manhattan assumption' based approach was also successfully demonstrated to improve dense, plane-based reconstructions on multi-view stereo data [42] and LiDAR scans (see Zheng et al. [192]). In another attempt, Straub et al. [154] proposed a probabilistic framework customized to describe scenes as a superposition of Manhattan frames and use hybrid Gibbs sampling with gradient-based optimization to abstract urban scenes. To better cope with clutter that would induce large numbers of frames, they only consider frames supported by more than 15% of all normals in the scene, effectively narrowing the method to work best for scenes with six or fewer dominant frames.

Boulch et al. [13] proposed an interesting discrete optimization method, that expresses edges and corners as high-order selection potentials on voxel grids to better abstract man-made environments, such as buildings. The approach, however, assumes dominant voxel directions similar to Manhattan frames, and is not easy to extend to scenes with multiple clusters of relations (*e.g.,* Figure 2.1). Xiao and Furukawa [179] proposed a CSG based method that models free space using 3D cuboids. Similarly to our method, they also first propose a super-set of possible orientation candidates. They then use a greedy algorithm to incrementally expand their encompassment of free space. The method takes an important step towards modeling non-local relations between planar primitives by being able to traverse continuous free space with the primitives. One of their main contributions relies in the observation that a point sample made by a laser scanner also yields information about the occlusion-free nature of the light path connecting the sensor and the sampled surface. In smaller environments and cluttered real world scenes the light paths are much more frequently interrupted and thus their incrementally built collection of primitives fractures into multiple orientations the same way as other methods assuming local continuity of surfaces. The main challenge is to balance between data fit and scene regularity, while still allowing small characteristic directions to retain their identity. Chapter 3 specifically targets such cluttered scenes, where all the points should be explained, and many relationships between primitives can be switched on or off, depending on the situation (*e.g.,* modeling pentagons).

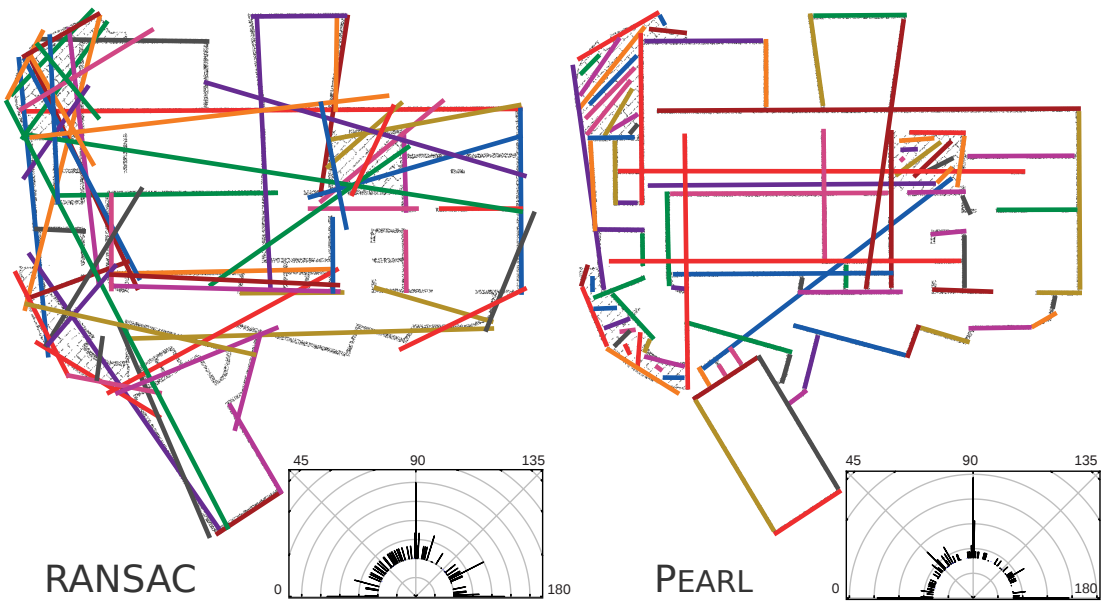


Figure 2.1: Regularization of primitive approximation of raw pointclouds by reasoning about regularity of orientations in man-made scenes. I converted a rasterized image of a floorplan to create a 2d pointcloud (shown in Figure 3.4). Images show extracted lines from the input pointcloud and the angular distributions of their orientations. Similar colors mean near-perfect relationships, where the set of sought after relations was specified in terms of their relative angles as $\{0, \pi/2\}$. Only a fraction of inter-line regularities are found in the noisy data using existing methods.

Indoor scene analysis. Acquiring and understanding indoor scenes are valuable for various applications. A common approach is to design a classifier to label scene objects [141, 181]. Silberman et al. [150, 151] reported good accuracy performance in indoor scene labeling using learned probabilistic discriminative models for data and compatibility terms from an RGBD image dataset and by considering support relationships. In another research thread, interactive methods have been developed for 3D modeling and content-aware image editing. Shao et al. [145] segmented RGBD images with semantic labels, and retrieve 3D models corresponding to the labels from a database using a random regression forest for the depth data. In concurrent efforts, Kim et al. [81] developed a model-based algorithm to group the extracted geometric primitives from depth data into individual models, while Nan et al. [111] interleaved segmentation and classification in region growing to extract objects from depth data. The idea about exploiting intra-scene regularity in man-made scenes has also been applied specifically to office environments [82]. Chen et al. [19] exploited scene context to recognize objects in an attempt to rapidly consolidate large scale interior RGBD

images. More recently Mattausch et al. [100] proposed an unsupervised algorithm to identify and consolidate repeated objects across large office scenes.

Abstraction, local fitting. Many shape analysis algorithms attempt to cope with noisy input data by robustly fitting simple and approximate primitives to raw pointclouds, *e.g.,* planes or cylinders. The most popular approach involves progressively fitting primitives using RANSAC [142] and its many variants (for example [21, 115]). While these methods are often robust to noise, recovering inter-primitive relations is not the goal of such local statistical methods (see Figure 2.1). In the context of 3D meshes, Mehra et al. [102] proposed to abstract man-made objects by arrangement of feature curves. The work was later generalized to co-analyze collections of shapes to produce mutually consistent abstractions across shape collections [187]. In the context of urban modeling, strategies involve integrating information from multiple acquisition modes [192], combining images and scans [91], or allowing user interaction to indicate symmetry structures from 3D data (*e.g.,* SmartBoxes [110]). Lin et al. [93] proposed a hierarchy-tree based system to perform semantic decomposition and large-scale, progressive reconstruction of urban LiDAR scans. In a very interesting effort, Sharf et al. [148] proposed so called mobility-trees to capture high-level functional information in man-made scenes. The extracted information encoded scene-level object information along with their part-level motion attributes. Very recently, a novel proactive scanning algorithm [184] was proposed to allow interaction with the scanned objects in order to acquire and combine recordings of visible and less easily visible object parts.

Ataer-Cansizoglu et al. [7] proposed an RGBD based SLAM method for utilizing the compact representational power of planes to render real-time localization algorithms even more computationally efficient. They incrementally build their planes by assigning new observed points to previously identified planes and iteratively refitting the parameter estimates. Due to the construction process being an online method, points belonging to the same planar primitive are required to be spatially continuously distributed, there is no option of identifying non-local relations between planar primitives. In follow-up work, Salas-Moreno et al. [135] presented a method that identifies the shared orientation of spatially proximate surfels and achieves impressive representation compression ratios, resilience towards discontinuities in sampling due to the hierarchical nature of their representation consisting of planes and surfels, and high regularity among densely

sampled surfaces. Despite the hierarchical nature of their formulation, the method does not target to identify non-local relations between planar surfaces, which would grant them additional robustness to sampling noise and tracking errors. Similarly to our formulation, they also dedicate special attention to non-planar surfels, which allows them to not over-regularize their scenes.

Constrained data fitting. As an alternative, the GlobFit framework [90] detects a set of relations from an initial set of primitives (obtained by RANSAC), and then performs data fitting in a constrained optimization setup. However, in such a two stage approach, if the initial set of extracted primitives is corrupted, erroneous relations are easily introduced, and worse 'conformed' to in the fitting stage. In another attempt, Lafarge et al. [87] introduced point-set structuring for first consolidating raw inputs and resampling extracted planar components. They then propose a Delaunay triangulation based hybrid reconstruction to produce high-quality reconstructions. The method cleverly combines canonical parts from extracted (planar) primitives and free-form parts of the inferred shapes. More recently, Oesau et al. [118] propose a graph-cut based formulation for abstracting primitives in indoor scenes. Ramalingam et al. [128] proposed an efficient method for extracting an arrangement of 3D lines from a single facade image using vanishing points and orthogonal structures via an interesting LP-based optimization. Such methods, however, decouple primitive extraction from relation detection, which is what the method described in Chapter 3 is trying to avoid. In more interactive contexts, Sinha et al. [152] allowed users to interactively model buildings by annotating over images. Arikan et al. [5] allowed the user to interactively specify relations and connections among the set of initial primitives extracted by RANSAC, with the system simultaneously fitting to the input point cloud and ensuring planarity of the polygons.

In a more general data analysis context, the PEARL setup [71] formulated model-fitting as a labeling problem using a global energy formulation. However, the global coupling involved a smoothness prior across neighboring primitives and hence, did not consider broader inter-primitive relations, which are dominant in man-made scenes (see Figure 2.1b).

Pham et al. [122] extended this framework to pairwise non-spatial relations. To preserve diversity in already medium-scale scenes remains difficult using their method due to the exponential explosion of conformity suggesting relations. In comparison, the

solution presented in Chapter 3 focuses on simultaneously extracting the primitives and non-local relations in a coupled, global and robust formulation.

Rother et al. [131] presented an efficient optimization technique to discrete labeling problems containing higher-order energy functions often used to model relations between groups of represented entities. Their regularization through cliques concept inspired our proposed method in Chapter 3. In follow-up work, Ladicky et al. [86] show how higher order potential fields can be interpreted as encodings of co-occurrence statistics. Optimizing for regularity of orientations in a reconstructed scene can be interpreted as optimizing for a type of co-occurrence, where exactly matching orientations are designed to much more likely appear in the output reconstructions.

Shape completion. Occlusion naturally results in missed data. Various approaches have been proposed to plausibly complete such missing regions: diffusion-based smooth surface completion [25]; example-based completion based on existing database models [147, 121] and local context-based completion [56]. Since finding an appropriate shape from model repositories is often ill-posed especially in absence of sufficient data, Shen et al. [149] introduce a structure-based approach to extract suitable model parts and fuse them together to form high-quality models. Note that in these cases the resultant model, retrieved as a whole or as an assembly of parts, remains plausibly connected based on the connectivity inherited from the source models. Such methods tend to fail in presence of significant clutter and occlusion.

Proxy-based scene analysis. Instead of assuming access to all possible scene objects, an alternative approach is to explain the input in terms of ensembles of simple proxies. The recovered network of relations among the proxies can be considered as a form of 3D scene understanding. Popular approaches beyond the aforementioned methods [90, 87, 5] involve encoding input scenes as collections of planes, boxes or cylinders, and studying their spatial layout [53, 88, 58]. In the context of image manipulation of indoor man-made environments, Zheng et al. [193] abstract scenes with cuboid primitives to facilitate simple yet intuitive edits. These methods, however, do not yield information about the occluded scene parts, which is the focus of the work presented in Chapter 4. Gupta et al. [55] reason about object layouts based on human interactions in typical workspaces. In order to improve accuracy of cuboid proxy detection, Fidler et al. [33] learn a statistical deformable cuboid model, whilst

Xiao et al. [180] learn a cuboid corner point model to better detect corners in images, eventually leading to better cuboid models. Hedau et al. [61] design a set of image space contrast-based features to better fit proxy boxes to explain scenes from single views.

Physical constraints, such as penetration-free, are also adopted when reasoning about the position of proxies in an indoor scene [62], while Umetani et al. [165] use physical stability and torque limits for guided furniture design in a modeling and synthesis setting. In parallel efforts, Kim et al. [78] jointly estimate a voxel-based 3D reconstruction along with voxel labelings using a voxel-CRF formulation. Physical validity constraints [73, 74, 191] have also been used for voxel-based scene parsing. The methods mostly focus on the physics of static scenes and use local reasoning on physical stability, which by itself is not sufficient to ensure global stability (*e.g.,* all three blocks in Figure 2.2a have to be simultaneously optimized to produce a stable stack).

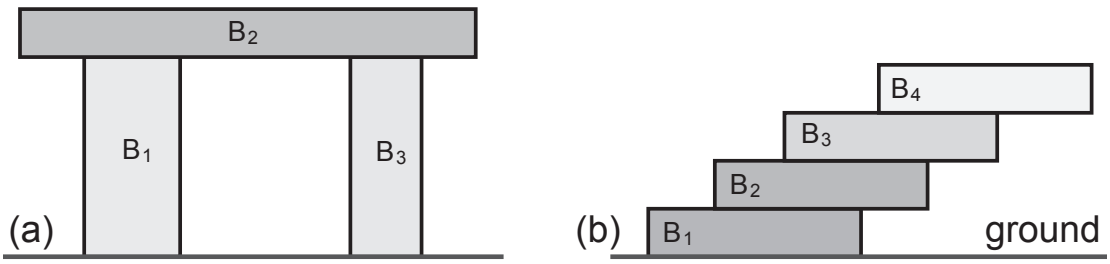


Figure 2.2: In order to evaluate physical stability, simply investigating pairwise proxy interactions is insufficient. (Left) Although each pair of touching boxes is by itself unstable, the arrangement of all the boxes is stable. (Right) Although each pair of touching boxes is stable, the arrangement of all the boxes is unstable.

Chapter 4 also focuses on proxy-based scene understanding, particularly considering global physical stability to reason about large scale missing geometry, but without access to rule-based priors or any training data.

With further advances in acquisition technologies, several large-scale indoor reconstruction datasets have been created [18, 23]. Starting from similar 3D scene collections, several previous works focus on analyzing inter-object relationships [35, 190, 64, 63, 182] and hierarchical grammars [94]. Another line of work recovers layouts from single images [137, 125, 72, 162, 69] by matching individual 3D objects. Relationships between the matched objects have been used to further regularize the recovered layout [143], or a collection of primitives to reconstruct accurate room geometry from images [26]. While the goal in Chapter 6 is to also recover an approximate

3D scene layout of a partially observed scene, the method relies on detected human interactions to reason about occluded objects.

2.2 Analysis of dynamic scenes

Human-centric shape analysis. Earlier work that uses observations of how humans interact with objects focuses on tasks such as object and event recognition [54, 27, 173] and action detection [185]. Kim et al. [79] propose a shape analysis tool based on a human-object affordance model that can be used for many applications including correspondence estimation, shape retrieval, and view selection. Subsequently, Fu et al. [40] utilize a similar model to generate new objects by combining functional parts of existing objects, while Pirk et al. [124, 123] introduce the concept of *interaction landscapes* as a descriptor of an object based on the type of interactions it can be involved in. More recently, Gkioxari et al. [47] predict human-verb-object instances from a single image to characterize human-object interactions. These methods focus on individual objects and their (potential) interactions. In comparison, the goal of the method in Chapter 6 is to use partially observed human motion to recover a full scene and a consistent human performance.

Dynamic geometry acquisition. In case of dynamic scenes, even state-of-the-art RGBD based systems [105, 113, 28] struggle with highly dynamic motion and occlusion. Capturing 3D geometry of dynamic scenes remains very challenging. For dynamic scenarios, inverse methods with physics priors have also been used to capture other phenomena such as liquids [169] or smokes [49].

The seminal system RAPID [57] represents a milestone for model-based tracking. First, edges are detected in the input image and the rendered model to increase robustness to view-dependent appearance features. Then, correspondences of point samples of the edges are sought and finally, the new parameters of the motion model are computed to estimate the updated state of the tracked object. Zisserman [195] renders the method more robust by adding a RANSAC step before the parameter estimation step, which enables the method to ignore falsely detected edgels. Pan et al. [120] proposed a model-based tracking method ProFORMA extending the above to apply incremental Delaunay tetrahedralisation to build the model of the tracked object in an online fashion. We are much looking forward to the further improvement of accessible ultra high-speed

capture devices that will allow the blur-free capture of appearance features that will enable methods to build their 3D reconstructions in our setting literally “on-the-fly”.

While impressive results have been demonstrated in case of template-based solutions for human faces [44, 2, 197], hair [183] and body [133], they focus on application specific contexts where object behavior and dynamics can be captured and learned in a training phase. In contrast, the work presented in Chapter 5 focuses on reconstructing 3D geometry of collisions directly from raw videos. Note that due to the nature of the problem, collisions happen prohibitively fast for accurate capture. Furthermore, object parts are severely occluded near the actual collision. Results demonstrate, that motion states of the objects as well as internal physical properties can be estimated by analyzing the objects’ pre- and post-collision trajectories away from the time of collision.

Rigid body simulation. Rigid body simulation has a long and successful history within computer graphics. Starting with the early works in animation [6, 9], it is now widely used in all forms of computer simulation. A good overview can be found in the book by Eberly [29]. While the forward simulation problem is far from trivial, it is well studied. 3D scene analysis of dynamic scenes targets the inverse problem, and in the following I restrict the discussion to the relevant works.

The first approaches to edit or modify simulations proposed spacetime methods to compute optimal motions in constrained systems [178, 95]. Variants with neural networks [50] and genetic algorithms were proposed [156] to reduce the complexity or synthesize directable motions. A randomized approach to re-construct motions for target configurations of rigid bodies was proposed by Chenney et al. [20], while an interactive variant was developed by Popović et al. [126]. O’Sullivan et al. [119] proposed metrics to measure the visual fidelity of animations acknowledging the necessity for accurate dynamics in animation. Others have used the complexity of collision events to randomize the solution space, in this way giving control over the outcome of a simulation [164]. Recently, Smith et al. [153] investigated the even tougher case of multiple, simultaneous collisions. While I focus on the two body case, these works highlight the complexities of collisions, which are only amplified for imperfect real-world objects. A good general survey of collision modeling is the report by Gilardi and Sharf [46]. While the above algorithms offer varying levels of control and parameter estimation for rigid bodies, they focus on virtual situations in a simulator. The method proposed in Chapter 5 on the

other hand, works robustly with only sparse, unreliable data from a real world input source. Areas that would be highly interesting, but which I have currently not taken into account are deformable objects [159, 99] and fracture [109, 155]. In the following, I assume that the observed objects do not deform or change topology.

An interesting general direction of research investigates the retrieval of modeling equations from data by employing learning techniques [12]. While the proposed method in Chapter 5 is not directly using machine learning, it represents a *data-driven* approach to recover complex real-world phenomena. In this context, it demonstrates how suitable physical models are powerful regularizers for such underdetermined problems.

Conotter et al. [22] analyze the kinematics of the ballistic motion of a single object in videos in the digital forensics context. Salzmann and Urtasun [136] explain the change in the linear motion of objects by solving for a sparse set of forces, applying their dynamics motivated regularization in video based tracking. The work presented in Chapter 5 focuses on acquiring the intrinsic physical properties of the objects to guide the motion estimates taking both linear and angular motion into account.

Human pose estimation. With the recent success of deep learning, we have seen advances both in 2D [161, 114, 70, 174, 16] and 3D pose estimation [65, 194, 158, 160, 130]. In particular, the recent VNect system [104] demonstrates impressive world-space pose estimation from monocular video. Many of these approaches, however, do not specifically tackle the occlusion problem and fail under moderate to heavy occlusion. Only a few pieces of existing work focus on predicting human pose, either 2D [39] or 3D [66, 176], in the presence of slight occlusions from static input images. For example, Wei et al. [175] leverage user assistance to handle moderate occlusions. Such methods do not reason explicitly about occlusions arising due to human-object interactions, nor use interaction priors to recover occluded interactions. In contrast, the work in Chapter 6 utilizes the initial human pose estimates from state-of-the-art human pose detectors to jointly reason about scene layout and human pose to synthesize both consistent scenes and human motion even in scenes with moderate to heavy occlusions.

Human-centric scene synthesis. Inter-object relationships discovered during the aforementioned human-centric scene analysis can then be used to synthesize new scene variations, *e.g.,* by replacing objects or scene parts with those in different scenes such

that existing relationships are maintained [189, 67], using a Markov Chain Monte Carlo based approach [186], or using a probabilistic graphical model [34]. More recently, deep learning methods have been proposed to progressively synthesize consistent scenes [171]. These methods, however, require full knowledge of the 3D scene layouts instead of attempting to recover them from images or video footage. Recent efforts in scene synthesis incorporate human actions into scene analysis. Datasets of typical human actions have been used to infer where specific actions can take place in a scene [138], to regularize scene synthesis [98, 41], or to regularize reconstruction of a scene layout from incomplete 3D scans [36, 75]. Although these methods use models of typical human actions as priors, they do not explicitly use action or motion cues as input, and thus fail to recover occluded objects.

The work in Chapter 6 is inspired by methods that use human action or motion cues to reason about the scene geometry. Fouhey et al. [37] combine human pose estimates with appearance and other geometric cues to estimate the room cuboid and free (walkable) space inside the room from a single image or a time-lapse video. Frank et al. [38] recover an object layout from a manually defined set of human actions. Object types and shapes are inferred from specific motions that the human performs, like tracing the edges of a table with the hands. Objects are recovered by observing human motion during interactions, however, the types of motion that the actor needs to perform preclude working with natural videos.

The recent work of Savva et al. [139] analyzes interaction snapshots, *i.e.,* action and pose labeled RGBD sequences to learn *prototypical interaction graphs (PiGraphs)* to link attributes of the human pose to the surrounding objects. They show how PiGraphs can be utilized to generate scenes that correspond to *static* interaction snapshots (*e.g.,* lie on bed). Kang et al. [76] focus on a similar goal of scene synthesis that explores motion cues. However, their input is an occlusion-free 3D human motion sequence in contrast to my target scenes with moderate to severe occlusions.

Alayrac et al. [3] proposed a method to automatically identify actions and the range of possible object states in connection to those actions from input videos of those actions being performed. Starting from pre-trained object detectors, the analysis is performed in a joint fashion. They simultaneously look for the times the actions are performed and localize the target objects of the actions in space-time. Similarly to our method in

Chapter 6, the key insight is that the state change of the object aids the discrimination between, on one hand, the relevant action, and on the other hand, motion occurring uncorrelatedly. Simultaneously, the identification mutually aids the detection of the target object and the identification of its different states.

Li et al. [92] proposed a method to estimate the pose of a human interacting with an object from RGB video. They first detect the 2D joint positions, then solve an elaborate optimization problem to automatically compute the parameters of the necessary contact forces and the 3D motion of the human and the object. Similarly to the above referred solutions, the method does not handle medium and heavy occlusions.

I presented an overview of competing methods performing 3D scene analysis with the help of auxiliary prior knowledge at complexity levels ranging from abstraction by planar primitives through physical plausibility to human interactions. I now continue to present the proposed solutions that employ novel, not directly observable cues to improve model fitting estimates.

Chapter 3

Regularity in raw scans as non-visual cue

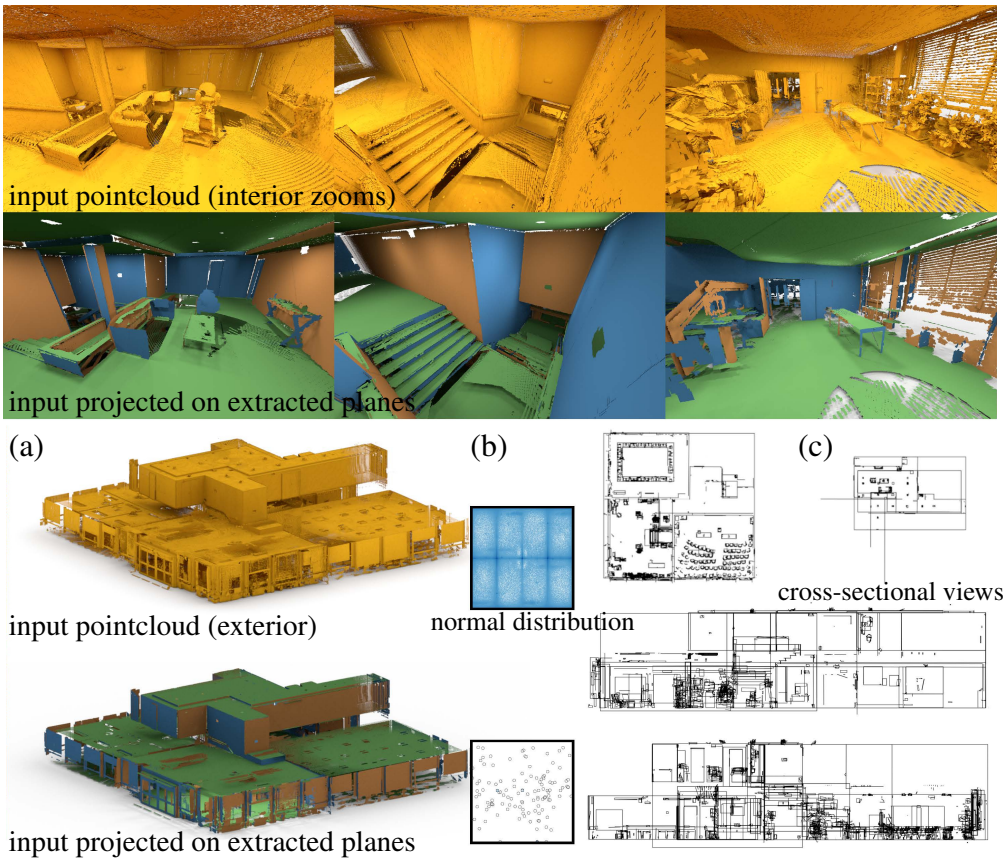


Figure 3.1: We present a novel approach to extract Regular Arrangements of Planes (RAP) from an unstructured and noisy raw scan (shown in gold). (a) In this example, our algorithm generates an abstraction of the raw pointcloud, pre-assembled from multiple laser scans. (b) The distribution of the initial normals is very noisy, which makes any greedy arrangement of planes error-prone. Instead, we propose a global algorithm to simultaneously select both the planes along with their sparse inter-relations. (c) Cross-sectional views reveal the discovered regularity of the extracted arrangements at multiple scales, *e.g.,* walls, stairways, and chairs. Parallel planes have same color.

3.1 Introduction

Pointclouds are now easy to acquire. They can be recordings of both indoor and outdoor environments, and can easily contain millions of samples. Such data volumes often retain interesting information about the captured scenes, and hence are keenly investigated in the context of scene understanding. These analyses generate valuable scene priors for various computer graphics applications. For example, they can reveal typical object arrangements in scenes [82, 100, 19], provide non-local priors for scene completion [192], deliver workspace affordance metrics [148, 184], or aid autonomous navigation by providing scene maps [4]. A grand goal is to abstract such massive data volumes [117, 116] to eventually produce semantic understanding of the scenes [13].

In the context of data analysis, one interpretation of abstraction [187, 87, 118] is to reveal interesting high-level global scene characteristics, rather than focus on local details, for example obtained via a full surface reconstruction. In the context of man-made scenes, we observe that scene characteristics are often encoded in the form of inter-part relations, inside and across objects. As many objects primarily comprise of planar faces, man-made scenes can be well abstracted as collections of planes, more importantly, planes along with their inter-relations. In this chapter, we focus on the problem of reconstructing raw pointclouds as regular arrangements of planes. The resultant abstractions provide compact and simplified representations, and expose high-level structures hidden in the raw data.

In the case of noisy, incomplete, outlier-ridden data, the challenge is to balance between compliance with data, and reliance on relations to regularize the extracted planes. However, one source of difficulty is that configurations of inter-plane relations are often scene-specific due to sensitivity to the primitives' local and global environments, and hence are not known *a priori*. They can easily be missed, or even worse, wrongly identified. The same set of 3D points can be scans of two parallel walls, or a wall and an ajar door depending on the context. Specifically, it can be particularly difficult to retain non-dominant plane orientations as they can easily be masked and falsely 'regularized'. *E.g.*, Figure 3.1 shows a scan of an entire building consisting of few million points and having details ranging from exterior walls, interior features such as staircases or rooms with scattered chairs, thus exhibiting a range of different relations in various parts of the

scene. The challenge is to detect these multiple interesting scene features in a unified setting.

The most popular pointcloud reconstruction approach is to progressively extract best fitting primitives using RANSAC, or its variants [142]. The method is attractive given its simplicity, scalability, and probabilistic guarantees. However, such a local and incremental analysis easily misses global scene-level structures (see Figure 2.1). Various refinements have been proposed including allowing users to explicitly annotate [5] and greedily identify inter-primitive relations [90]. The challenge is to bootstrap the algorithms, which all heavily depend on the initial set of primitives. As an extreme case, the initial inter-primitive relations in Figure 3.1 being extremely noisy, it is very easy for a greedy approach to make erroneous early commitments, resulting in significant overall degradation (see Figure 3.7).

In this chapter, we propose a non-local approach to *simultaneously* select a set of planes along with their relations. We make the key observation that global relations between planes persist over long distances in man-made scenes, counteracting some of the harm caused by noisy data. Hence, instead of extracting individual primitives, we extract sets of primitives, and in the process favor specific arrangements of primitives over less regular ones. We seek out regular arrangements with regularity being measured by aggregating agreements between primitive pairs in the extracted set. We enable the user to specify for a given task a certain family of sought relations, and our output allows the user to infer further relations. For example, in Figure 3.1, we extract a *regular arrangement of planes* (RAP) with mutually parallel/orthogonal primitive pairs (walls, chairs, and staircases), even at the expense of the resultant data fitting error being marginally higher. By working directly in the space of relations, the algorithm provides non-local coupling and allows reliable planes extracted in less noisy regions to influence and regularize the corrupted regions of the scans.

The algorithm starts by producing an initial set of candidate primitives using local analysis. Then, we generate a larger set of candidate primitives based on potential inter-primitive relations. This step effectively hypothesizes possible relations and allows reliable primitives to create potentially good candidates in less reliable parts of the data. Finally, in a key selection step, we extract a regular arrangement of planes by balancing between explaining the pointcloud and producing a simple and compact arrangement

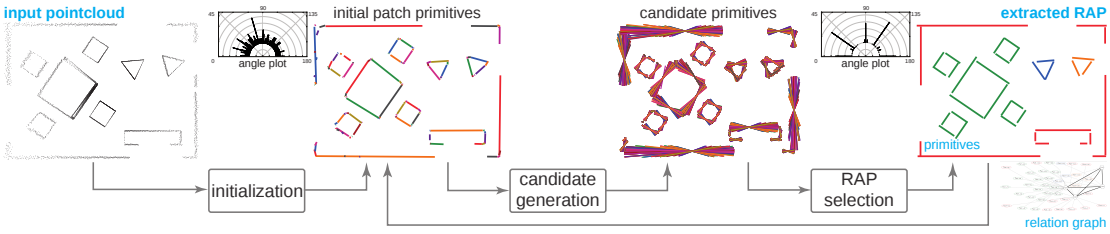


Figure 3.2: Algorithm overview. We present RAPTER to abstract a raw pointcloud by extracting a globally-coupled regular set of planar primitives (line segments in 2D) along with the inter-primitive relations. The method works in three main stages: (i) initialization, (ii) candidate generation; and (iii) regular arrangements of planes (RAP) selection. As an outer loop, we repeat this following a coarse-to-fine strategy with primitives at bigger scales regularizing extraction in regions of low confidence at smaller scales. Insets show the distribution of angles among normals of pairs of oriented points using the initial and final line segments and point-to-line assignments. Our method discovers the regularity among the lines, and enables weakly supported orientations retain their independence. Input angles: $\Theta = \{0, \pi/3, \pi/2, 2\pi/3\}$.

of planes. The main insight is to defer the final selection to the end, and extract the simplest globally consistent arrangement of planes that best explains the raw input. Thus, expanding the candidate set eventually simplifies the problem. As an important technical contribution, we formulate the RAP extraction problem as a mixed-integer program that allows multiple orientation relations to coexist, even when significantly unbalanced in corresponding number of witnesses. For example, in Figure 3.1, the chairs are still abstracted as sets of extracted planes, and not unduly regularized by big exterior building walls, *i.e.,* the chairs are not 'snapped' to align with the walls.

We evaluated our algorithm on a range of 2D and 3D datasets and found the method to be robust under noise, sampling variations, and missing data. The extracted primitive arrangements directly provide an abstraction of the input and significantly out-performed specialized structure analysis alternatives. In summary, our main contributions are formulating the problem of pointcloud abstraction as an instance of coupled selection among candidate plane primitives; and proposing a simple, robust, scalable algorithm to extract such a globally coupled RAP directly from raw pointclouds.

3.2 Overview

Our goal is to convert a raw pointcloud from a scanned scene into *regular arrangements of planes* (RAP), where regularity refers to a prescribed list of favored inter-plane relations. As output, we produce arrangements of planes, with associations to their respective raw points. See Figure 3.2 for an overview.

We observe that man-made environments primarily consist of planar parts that are mutually related. Typical inter-plane relations include parallelism, coplanarity, orthogonality, and symmetry. Hence, we give preference to such regular arrangements, to reconstruct the input pointclouds (*e.g.,* LiDAR and Kinect® scans). We measure goodness of fit as a balance between two factors: (i) *data cost:* the data-fit residual for approximating a set of points by its planar primitive, and critically, (ii) *irregularity cost:* the irregularity of the mutual arrangement of the output planes. A conventional *spatial* term ensures segmentation smoothness.

To achieve this, one simple approach is to first fit a set of planes to approximate the input data, then try to discover potential inter-plane relations, and finally, conform to the extracted relations using a constrained-fitting approach. This, however, assumes that the extracted relations are mutually consistent (*i.e.,* do not contradict each other). Alternatively, one can progressively build a set of consistent (potential) relations by selectively adding relations one at a time. Such a greedy selection strategy can easily result in a catastrophic failure by committing to an erroneous relation early on. Instead, we formulate a global optimization to *simultaneously* extract a set of primitives along with their relations. Our main observation is that instead of committing to any solution in the early stage, and hence introducing a bias, it is better to defer decisions to a later stage.

Actual inter-plane relations in the real-world are often concealed by flaws and bias in the acquisition stage. Hence, we first hypothesize possible relations and create additional primitive planes as speculative explanations for the raw points. In other words, we first *expand* the set of candidate primitives by adding hypothesized planes, and then repose the RAP extraction problem as a *selection* problem. Surprisingly, first significantly expanding the candidate set of planes and then formulating a constrained optimization to extract the RAP results in a simple, robust, and scalable algorithm.

The algorithm proceeds in three simple stages: First, starting from a point set $\mathscr{S}$, as *initialization* we generate a set of primitive planes $\mathscr{P} := \{P_i\}$ by locally fitting planes to the input. Then, in the *candidate generation* stage, each pair of planar primitives creates additional candidate planes that are added to the primitive set $\mathscr{P}$ to form the enriched set $\tilde{\mathscr{P}}$. Finally, in the key *RAP selection* stage, we formulate an energy minimization to select the RAP from the enriched set $\tilde{\mathscr{P}}$ as the final abstraction of the input data $\mathscr{S}$.

This results in the simultaneous extraction of primitives along with their inter-relations. We run the algorithm over coarse-to-fine scales, allowing the primitive arrangements selected in the coarse scales to influence and regularize the solution at finer scales. We describe the core algorithm next.

3.3 Algorithm

Starting from a pointcloud $\mathscr{S}$ of a man-made scene, our goal is to abstract the scene as regular arrangements of planes. As part of the input, we expect to know a priori whether certain inter-plane relations, such as 45°, contribute to a user-specified definition of "regular". The proposed algorithm runs in three main stages: (i) initialization, (ii) candidate generation, and (iii) RAP selection. RAP selection optimizes the balance between explaining $\mathscr{S}$, and imposing inter-plane regularity. We now describe these steps of the algorithm, and give more specific implementation details.

3.3.1 Initialization

We oversegment the point set $\mathscr{S}$ through simple region growing. The aim is to group nearby points, with consistent normals into patches. If needed, $\mathscr{S}$ is first turned into a set of oriented points by using local PCA analysis. The complete oversegmentation of $\mathscr{S}$ into a set of patches $\{S_i\}$ proceeds from the bottom up. Each point j within a Euclidean distance ρ of point i is grouped into patch S_i if its orientation differs from S_i's by less than τ. This process repeats, expanding the search-area to within ρ of any new point in the group. At each iteration, we compute the least-squares fit of a "local" plane P_i. These planes have finite extent, clipped to a bounding box based on the projection of points in S_i. We thus obtain our initial set of candidate primitive planes $\mathscr{P} := \{P_i\}$, see Figure 3.3a.

3.3.2 Candidate generation

The planar patches detected in the initialization stage are based on local fitting, and hence can easily be biased by noise and outliers (see Figure 3.3a). Robustly detecting relations and regularity among such noisy shape fragments is difficult, especially since bias may occur gradually over long distances. Instead, we explicitly generate additional planes as speculative "cousins", arranged relative to the initial set, according to the expected inter-plane relations. Most of these speculative patches will ultimately be

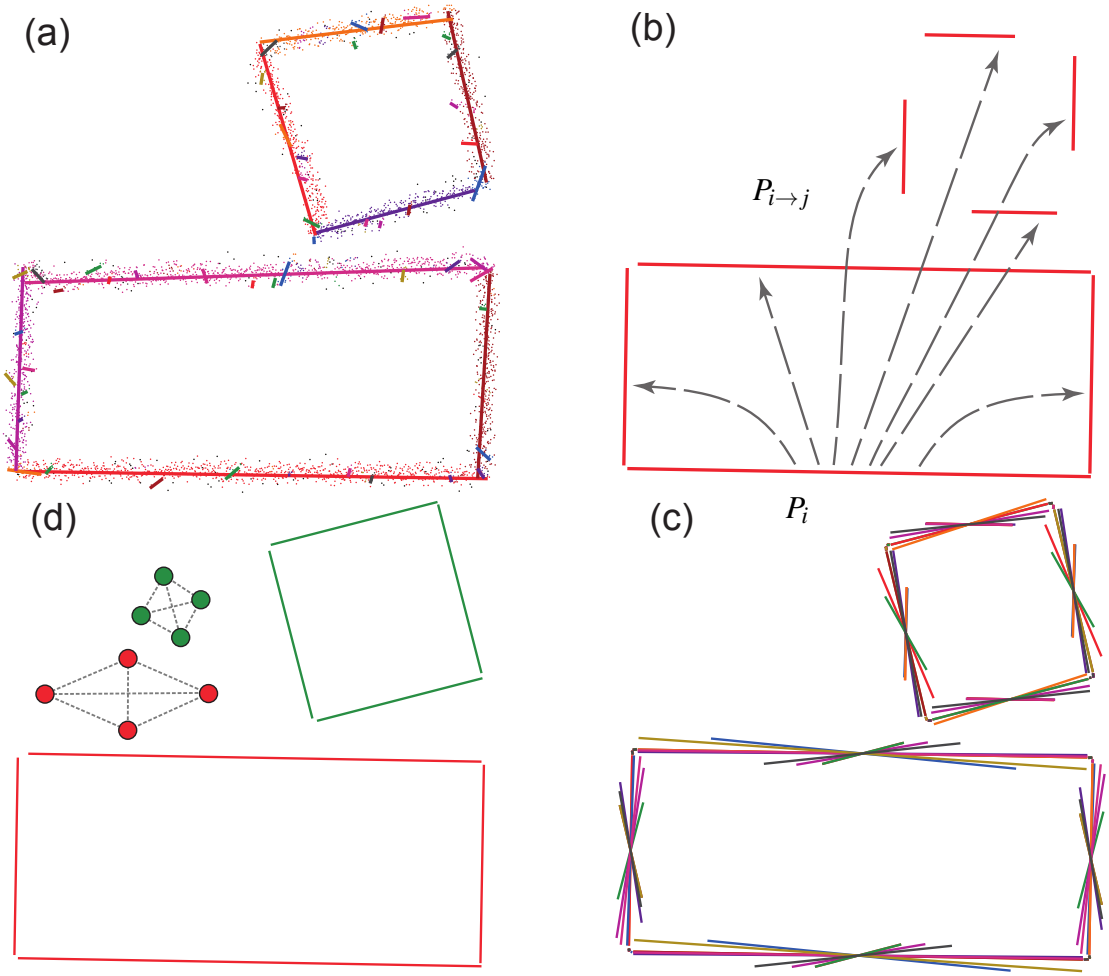


Figure 3.3: We oversegment an input pointcloud into (a) point patches S_i along with their planar approximations P_i. (b) Each such primitive generates candidate primitives at the centroid of other patches, i.e., P_i generates $P_{i\to j}$ at the centroid of any other primitive P_j, creating (c) a set of enriched candidates $\tilde{\mathscr{P}}$. (d) Finally, via a selection formulation, we extract a set of regularly arranged planes (RAP), i.e., a set of planes along with their relation graph (shown inset) by balancing between global regularity and faithfulness to the local data. Input angles: $\theta = \{0, \pi/2\}$.

rejected, but they help to recover some planes that were undersampled or noisy. Together with the initial planes, this expanded set of candidates allows us to pose the search for a good global RAP as a selection problem.

To be specific, each plane P_j in the initial set $\mathscr{P}$ was created to explain one patch of oriented points S_j. However, S_j may be better explained, in a global sense, by an alternative plane that comes from, *e.g.,* rotating plane P_i by 90° and translating it to S_j. We denote such a new plane as $P_{i\to j}$, and it serves as a candidate alternative to the initial $P_j \equiv P_{j\to j}$. Thus, each patch S_j is now associated with a set of candidate primitives $\{P_{1\to j}, P_{2\to j}, ..\}$, where $P_{j\to j}$ is an alias for P_j. We re-estimate the finite extents of

the other clones $P_{i\to j}$ based on the points in S_j. In practice, the number of possible inter-plane relations (i.e., $\Theta = \{0, \pi/2\}$) further multiplies the number of ways each P_i can be rotated onto S_j, but we keep the one with the best fit. The whole purpose of the candidate generation step is to ensure that the sought RAP configuration *exists* as a subset of the generated enriched candidate set $\tilde{\mathscr{P}} := \{P_{i\to j}\}$.

As an example, we visualize all the candidate subsets in Figure 3.3c, where the sides of the rectangles have proposed new candidates for all the patches in both rectangles. Figure 3.3b shows the generated candidates $\{P_{i\to j}, j = 1, 2, \dots\}$ from just one plane P_i. Details of how a user-selected family of plane relations is used to generate the enriched candidate set in a scalable way, are given in Section 3.4.

3.3.3 RAP selection

We are now ready to formulate the RAP extraction problem as a subset selection problem from the enriched candidate set $\tilde{\mathscr{P}}$. We represent the selection of any one candidate $P_{i\to j} \in \tilde{\mathscr{P}}$ by a corresponding binary indicator variable $\chi_{i\to j} \in \{0, 1\}$. The binary vector $[\chi_{i\to j}, \dots]$ corresponds to a selection of candidate planes, a RAP.

Formulation. We pose the RAP extraction problem as an energy minimization using these binary selection variables. We want to balance simultaneously explaining the scene in a data-faithful way, and selecting an as-regular-as-possible arrangement of planes.

We encode this as a weighted combination of three terms:

$$\{\chi_{i\to j}\} = \underset{\{\chi_{i\to j}\}}{\operatorname{argmin}} E := \lambda\ E_{data} + (1-\lambda)\ E_{irr} + E_{spat}, \tag{3.1}$$

where $\lambda \in [0, 1]$. To ensure that we assign each patch S_j at least one primitive candidate $P_{i\to j}$ to explain its data points, we require $\sum_i \chi_{i\to j} \geq 1\ \ \forall j$. For example, with $\lambda = 1$, all $\chi_{j\to j} = 1,\ j = 1, 2, \dots$, and the rest of the variables are 0, *i.e.,* only the original locally fit candidate planes get selected.

Data cost. Potentially, many planar candidates can be indicated as trying to explain each patch. We compute the total data fitting error as the sum of the individual data fitting residuals. With $E_d(P_{i\to j}, S_j)$ denoting the residual cost of abstracting patch S_j by $P_{i\to j}$, hence $E_{data} := \sum_j \sum_i \chi_{i\to j} E_d(P_{i\to j}, S_j)$, see Eq. (3.6).

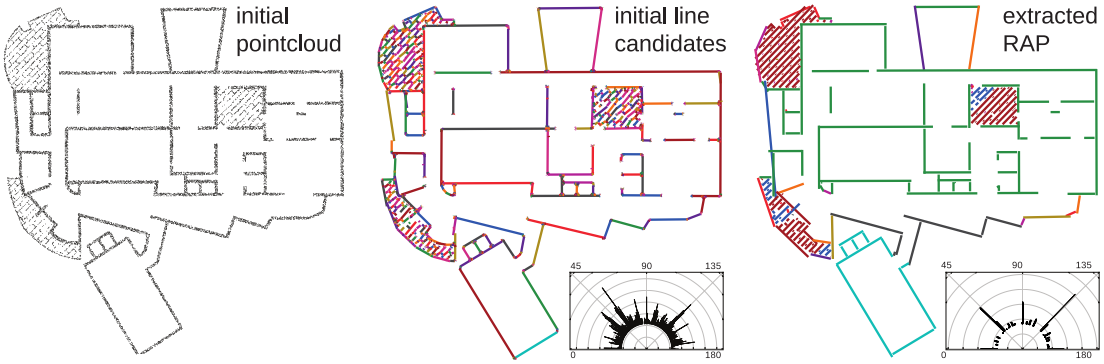


Figure 3.4: Starting from a pointcloud obtained by digitizing a floorplan our algorithm takes the initial line candidates to extract regular arrangements. Corresponding angular plots show the normal distribution of output points. Regularity is masked in the noisy initial plot (spread distribution), but is visible in the final extracted RAP (clustered distribution). Final line segments are colored based on their corresponding orientation frames, same color denotes lines mutually parallel or orthogonal (compare to Figure 2.1). Input angles $\theta = \{0, \pi/2\}$.

Irregularity cost. Even a bad choice of arrangements of planes may satisfy the constraints and have a low data cost. To also encourage *regularity* in the arrangement of planes, it seems natural to construct an undirected irregularity measure $Irr(\cdot,\cdot)$ between every pair of planes present in the proposed RAP. Similarly to Pham et al. [122], we formulated this as $E_{irr} := \sum_{j,i,l,k} \chi_{i\to j}\chi_{k\to l} Irr(P_{i\to j}, P_{k\to l})$, measuring the irregularity of any selected arrangement encoded by the indicator variables. By construction, certain pairs are perfectly compatible, for example, $Irr(P_{i\to j}, P_{i\to i}) = 0$, since both are generated from P_i using an a priori known inter-plane relation, hence these are automatically favored for selection.

Figure 3.3d shows an example of the selected RAP that prefer a perfectly regular arrangement, even at the expense of a slightly higher data cost ($\lambda = 0.5$ in this example). This is a desired behavior in such a balanced setting. Once regularized, a RAP can be visualized as a simple graph, with selected planes $P_{i\to j}$ as nodes. The non-zero indicator variables encode which inter-plane relations are part of the regularized RAP, as shown by the graphs in Figure 3.3d.

The scene in Figure 3.5a has its RAP depicted in Figure 3.5c, where the irregularity term has generated such pairwise potentials, that all 12 planes were interconnected perfectly. As a consequence, we see in Figure 3.5a that the data of the rotated square (data potentials not pictured) has been overruled by the axis-aligned majority. This is a common problem in scenes with large relative differences in the number of witnesses for the underlying orientations. The construed normalization variable in [122] is an attempt

to resolve this problem. However, even moderately complex scenes like Fig. 3.2 require more control and respect for diversity than this allows for.

3.3.4 Irregularity revisited

In Figure 3.5a, we have just seen that the naïve irregularity term E_{irr} can be too aggressive in encouraging *all* planes to be related to each other. Similar unaligned objects in a Manhattan-world drove the work of Straub et al. [154], which translates to RAPTER with input angles $\Theta = \{0, \pi/2\}$. We refer to a set of orientations that, given the input relations, are internally perfectly regular as a *frame*.

Here, we redesign the E_{irr} cost to tolerate diversity. Obviously, the clusters of planes or objects that should be regularized separately are not known in advance. So we

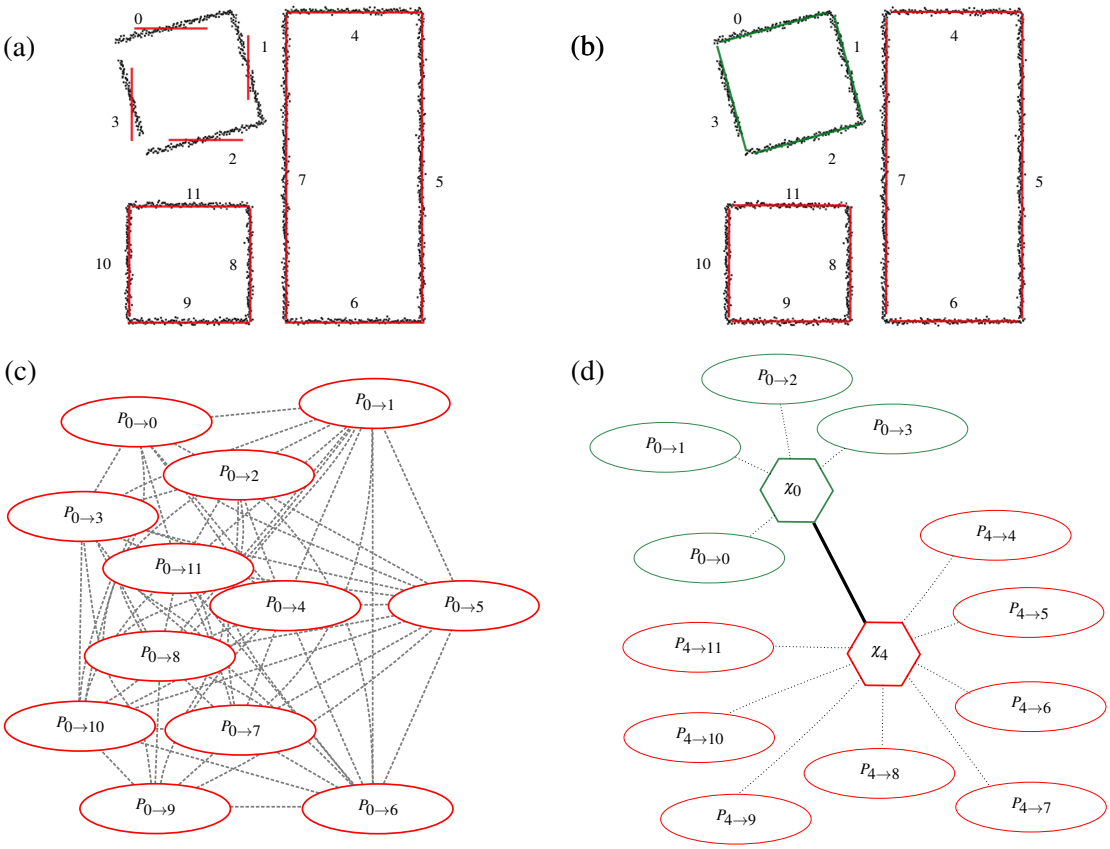


Figure 3.5: The direct pairwise formulation leads to an undesired over-regularization (a) as non-dominant orientation frames easily get masked by dominant ones. A single frame inspired by $P_{0\to0}$ gets selected (c), because the number of corresponding pairwise costs for across-frame connections relate exponentially to clique sizes. Our irregularity term (Section 3.3.4) can preserve distinct frames that represent only a minority of the points (b) in the pointcloud by introducing intermediate metanodes (hexagons) as shown in (d). Dashed lines and colors denote shared frames, the solid line denotes non-zero irregularity costs. Input angles $\Theta = \{0, \pi/2\}$.

introduce a second level of indicator variables. For each internally-regular coordinate frame, we add a binary auxiliary variable in the optimization $\chi_i \in [0,1]$.

Tolerable irregularities. The irregularity cost is intended to encourage abstractions that cope with low quality data and to align even distant patches. The regularization can, however, result in undesired simplification in areas with high data fidelity. All cross-cluster plane pairs contribute to E_{irr}, so a sole rotated table in a room is literally punished from all sides. If we fail to distinguish between irregular arrangements vs. clusters of regular arrangements, we end up penalizing both. Instead, we desire a reconstruction like Figure 3.5b. The new auxiliary variables $\{\chi_i\}$ signpost groups of planes that are regular among themselves, *i.e.,* where one χ_i represents its own frame. Note that even distant objects may be part of the same frame. We reformulate the irregularity energy so that the optimized RAP for Figure 3.5a looks like Figure 3.5b. With the new irregularity, we observe that the energy of each frame encourages mutual regularity internally, but different frames only pay the price for being misaligned once; see the bold edge connecting the indicator variables' hexagonal "metanodes." We redesign the irregularity term:

$$E_{irr^\star} := \sum_{i,k} \chi_i \chi_k Irr(P_{i\to i}, P_{k\to k}). \tag{3.2}$$

The original indicator variables $\chi_{i\to j}$ reveal which planes are selected as frames, so $\chi_i = \max_j \chi_{i\to j}$. Essentially, in the new irregularity measure $E_{irr^\star}$, the cost for choosing a new frame becomes less dependent on the number of similarly oriented primitives in the solution. If any of the candidate planes created from an initial locally-fitted plane in the candidate generation step is chosen, the corresponding auxiliary variable should get activated. We encode this behavior with the following constraints.

Constraints. An auxiliary node χ_i representing frame i contributes to the irregularity cost if any of the candidates $\{P_{i\to j}\}$ generated by the initial plane primitive $P_{i\to i}$ is selected. For each initial (plane) orientation i, we encode the max-condition as a single quadratic constraint of the form

$$\sum_j \left(\chi_{i\to j}\chi_i - \chi_{i\to j}\right) \geq 0 \quad \forall i. \tag{3.3}$$

Thus, we arrive at the final updated formulation as

$$\{\chi_i\},\{\chi_{i\to j}\} = \underset{\{\chi_i\},\{\chi_{i\to j}\}}{\operatorname{argmin}} \; E := \lambda\, E_{data} + (1-\lambda)\, E_{irr^\star} + E_{spat}, \tag{3.4}$$

where $\lambda \in [0,1]$, subject to the constraints $\sum_i \chi_{i\to j} \geq 1 \; \forall j$, and Equation 3.3. Note that the minimization produces the RAP abstraction, *i.e.,* both the selected set of planes and their inter-relations.

Spatial smoothness. Our main contribution comes from the global regularity of frames, where spatially distant objects are able to influence the regularity of the approximation. Since we start from an over-segmentation, we use a spatial smoothness term to encourage neighboring patches to pick the same plane orientation. We increment the energy with a fixed penalty for non-smoothness,

$$E_{spat} := \sum_{j,i,l,k;\; j\neq l} \chi_{i\to j}\chi_{k\to l} neigh(S_j, S_l) C_{spat}, \tag{3.5}$$

where the fixed spatial penalty $C_{spat} = (1-\lambda)/10$ is related to the irregularity weight. See Section 3.4 for estimation of patch proximity.

We need no additional term to limit the complexity of the arrangement (*e.g.,* number of selected planes $\sum \chi_{i\to j}$) as superfluous candidates get penalized by the total energy cost E. We proceed with details of the implementation and optimization.

3.4 Implementation Details

In this section, we provide various implementation details, comment on the optimization, and discuss necessary modifications to transition to 3D and for handling very large scenes. Although we detail the specific functions used, they could likely be replaced by comparable functions or cost measures leading to similar performance.

Initialization. For the initial oversegmentation of the input pointcloud, we progressively gather points if they are neighboring and have comparable orientations. We use the approach proposed in Yan et al. [184]. As angle threshold we use $\tau = \pm 15^\circ$ and a point distance of ρ. Apart from the list of possible plane relations, ρ is the main user-selected parameter of the system, and represents measurement-tolerance. Each new scene is scaled to fit the unit cube. Then ρ is set so that structures with size above

this threshold will be preserved, whilst other, smaller variations will be simplified. The resolution of the RAP can thus be modified according to the scene characteristics and application context.

Data term. We measure the data cost in terms of the total residual error of abstracting all patch S_j's points $\{p_h\}$ by the primitive plane $P_{i\to j}$, as shown in Equation (3.6). Each candidate plane has a fixed cost determined by its assignment to a patch. Our plane primitives are finite, its bounding box encloses all points p_h within ρ distance from the infinite plane. This means that when the point p_h, projected onto the plane, falls inside the patch's bounding box, the distance $d()$ is the typical point to infinite-plane distance. When p_h falls outside the bounding box, $d()$ is the Euclidean distance to the nearest point inside. Therefore, our data cost is

$$E_{data} := \sum_j \sum_i \chi_{i\to j} E_d(P_{i\to j}, S_j) \quad \text{where,} \tag{3.6}$$

$$E_d(P_{i\to j}, S_j) = \frac{1}{|\{p_h \in S_j\}|} \sum_{p_h \in S_j} d\left(p_h, P_{i\to j}\right)^2. \tag{3.7}$$

We normalize the per-patch residual error because patches have different numbers of points p_h. Point to finite plane distance provides robustness towards the smoothing effect of this normalization.

Irregularity. The most basic, yet surprisingly common regularity in an arrangement of planes is parallelism. We define irregularity as a function of the unsigned angle between the normals of two planes $Irr(P_i, P_j) := f(\angle(\mathbf{n}_i, \mathbf{n}_j))$, with $f()$ defined below. Regular arrangements contain planes with low irregularity, *i.e.,* the angle between their normals is close to 0. When the angle is 0, we think of this as a *perfect* relation. By construction, generated candidates are perfectly related with their generators.

Candidate generation. For each pair of planes $P_{i\to i}$ and $P_{j\to j}$, we generate two new planes. $P_{i\to j}$ comes from translating $P_{i\to i}$ to the centroid of S_j and rotating by one of the allowed relations (see Possible relations below), and similarly for $P_{j\to i}$ from $P_{j\to j}$. The coarse-to-fine iterations (see Scalability and Algorithm 1) mean that the output of a selection step serves as input for successive generation steps, *i.e.,* $P_{i\to j}$ can spawn $P_{i\to k}$ at patch S_k.

Pairwise term. Our main goal is to discover perfect relationships in the form of extracted RAP. In our implementation, we used $f(x) := 1 - \exp(-\delta x)$ with $\delta = 3$, to ensure that perfect regularity is encouraged, while still respecting minority frames.

Spatial smoothness. We use a spatial smoothness term to encourage neighboring patches to pick the same orientations allowing co-planar patches to be identified. Specifically, we check if the minimum distance between a pair of patches is small by estimating the minimum distance between their points, and if they have comparable orientation up to τ (angle tolerance), *e.g.,*

$$neigh(S_j, S_l) := \mathbf{1}\left(\min_{\mathbf{p}_g \in S_j; \mathbf{p}_h \in S_l} \|\mathbf{p}_g - \mathbf{p}_h\| < 2\rho\right) \mathbf{1}\left(\angle\left(\mathbf{n}(S_j), \mathbf{n}(S_l)\right) < \tau\right). \quad (3.8)$$

Quadratic programming. The RAP extraction problem amounts to a quadratic optimization problem with quadratic constraints. Note that the quadratic constraints are critical, as explained in Section 3.3.4 and Figure 3.5. We used a mixed integer nonlinear program solver [11] that relies internally on a modified (for nonlinear problems) branch-and-cut algorithm, and an interior-point based LP solver [168] for the relaxed sub-problems. We provided the solver with analytically computed second-order derivatives for the objective and constraint matrices. Note that the formulated constraints (and objectives) are non-convex, so the retrieved solutions are not guaranteed to be optimal. However, the quality of our output relies crucially on *robustly* retrieving a RAP, and we have not had problems with convergence in practice. Typically, for smaller problems, we ended up with 500 variables, while for larger problems, we had many more variables that we tackled by splitting the problem into groups of approximately 2300 variables, see Algorithm 1. We also used [134, 140] in our implementation.

Possible relations. Supporting other relations in RAPTER is relatively easy. In short, for any such relation one has to appropriately adapt the definition of regularity (irregularity in our case), and adjust the candidate generation step. In our implementation, the system supports planes that are

(i) Parallel: These are the default relation, require translation, and no further modifications to the energy terms.

(ii) Orthogonal: For irregularity, we simply measure the difference of angles between the planes from target angle $\pi/2$. For candidate generation in 2D, we translate the planes

to their target location (as before), and rotate by $\pi/2$. In 3D, it is a bit more involved. Given two planes P_i and P_j with respective normals $\mathbf{n}_i$ and $\mathbf{n}_j$, we create $\mathbf{n}_{ij} = \mathbf{n}_i \times \mathbf{n}_j$. Then, we set the direction of $P_{i\to j}$ as $\mathbf{n}_i \times \mathbf{n}_{ij}$, and $P_{j\to i}$ as $\mathbf{n}_j \times \mathbf{n}_{ij}$. The rest of the setup is unchanged.

(iii) Other angles: For other generator angles Θ^*, we proceed just as above, but use multiples of Θ^*, instead of $\pi/2$. As a result, we get the set of input angles $\Theta = \{k\Theta^*\}$, $k = 1,2,\ldots$; *s.t.* $k\Theta^* < \pi$. We adapt Irr to be $Irr(P_i,P_j) := \min_k \; f(\left|\Theta_k - \angle(\mathbf{n}_i,\mathbf{n}_j)\right|)$.

(iv) Coplanar: To model coplanarity, for any two planes P_i, P_j with the same (unsigned) orientation $\mathbf{n}_i = \mathbf{n}_j$, we set $Irr(P_i,P_j) := f(d(P_i,P_j))$ based on the offset distance $d()$ between the planes. Planes are simply mutually copied across in the candidate generation stage. In practice, other relations are explored together to optimize a RAP, that then seeds a coplanarity-only RAP, where we simplify (*i.e.,* merge) adjoining coplanar planes.

Scalability. For better efficiency, we make following modifications:

(i) In the candidate generation stage, we restrict candidate generation to pairs of planes with low to medium irregularity. Essentially, we put a threshold on the angle (or offset difference) to reduce the number of generated candidates. The same parameters are used, as for the initialization ($\tau = \pm 15^\circ$, see Table 3.1 for ρ).

(ii) We propose a coarse-to-fine RAP extraction workflow. First, at a coarse level, we only consider the larger initial planes (based on their area). Once we extract a corresponding RAP, we freeze these relations, *i.e.,* they are not allowed to change further. Then, we bring in the next level of patches, proceed as before, but allow the earlier RAP to also contribute in the candidate generation step and during selection. Note that we keep the older frame metanodes in place for the new patches. Essentially, relations detected at the coarse level can influence the ones lower down. However, the lower levels have no influence on the higher (*i.e.,* coarser) levels. This has proven to be effective because larger planes are more well-sampled. We go down in scale by factors of 2 as detailed in Algorithm 1. In line 16, we show one strategy to control the number of variables in the quadratic program by gradually introducing generated candidates. A strategy, that directly controls the number of metanodes (~quadratic terms) might be even more useful in practice.

Algorithm 1: RAP extraction by RAPTER

Input: Oriented points grouped into patches $\{S_j\} \in \mathscr{S}$, Local fits $\{P_{j\to j} \in \mathscr{P}\}$, relation generators Θ^*

Output: RAP $\mathscr{P}^* = \{P_{i\to j}\}$, Regular relations $\{\langle P_{i\to j}, P_{i\to k}, rels\rangle\}$, Point assignments $\{p_h \to P_{i\to j}\}$

```
/// (1) Initialization
```
$\mathscr{P}^0 := \{P_{j\to j}\} \in \mathscr{P}$ `// Sort initial fits by ↓ area`

$\mathscr{P}^* := \emptyset$ `// Initialize set of selected candidates`

$\theta := \frac{area_{max}}{area_{min}}\rho$ `// Estimate area threshold`

while $\mathscr{P}^0 \neq \emptyset$ **do**

```
    /// (2) Candidate generation
    // Select sufficiently large initial fits
```
    $\mathscr{P}_\theta := \{P_{j\to j}, s.t.\ \forall_j\ area(P_{j\to j}) > \theta\} \in \mathscr{P}^0$

    $\mathscr{P}^0 := \mathscr{P}^0 \setminus \mathscr{P}_\theta$

```
    // Enrich from coarser and same scale
```
    $\tilde{\mathscr{P}} := \mathscr{P}_\theta \cup$ `Enrich(`$\mathscr{P}_\theta$, $\mathscr{P}^*$`)` $\cup$ `Enrich(`$\mathscr{P}_\theta$, $\mathscr{P}_\theta$`)`

```
    /// (3) Selection from enriched set
```
    $\mathscr{P}^* := \mathscr{P}^* \cup \{\mathscr{P}^{*'} \subseteq \tilde{\mathscr{P}}\}$ `// Minimize Equation (3.4)`

    Simplify nearby co-planar patches

```
    /// (4) Iterate
```
    if *No break in* `Enrich` **then** decrease area threshold $\theta := \frac{\theta}{2}$

return $\mathscr{P}^*$

Function `Enrich(`$\mathscr{P}$, $\mathscr{P}_{fixed}$, $|\tilde{\mathscr{P}}|_{max} = 2300$`)`

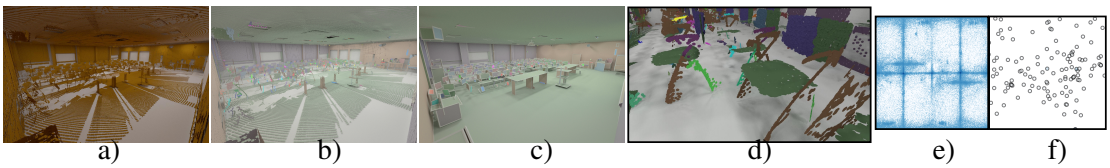
for *Each pair* $\langle P_{j\to j} \in \mathscr{P},\ P_{k\to l} \in \mathscr{P}_{fixed}\rangle$ **do**

    if $\min_r\left(\left|\Theta_r - \angle(\mathbf{n}_j, \mathbf{n}_k)\right|\right) < \tau$ `// Min angle`

    then $\tilde{\mathscr{P}} := \tilde{\mathscr{P}} \cup \{P_{k\to j}\}$ `// Create new candidate`

    if $|\tilde{\mathscr{P}}| > |\tilde{\mathscr{P}}|_{max}$ **then** break `// Limit #variables`

return $\tilde{\mathscr{P}}$ `// End of function Enrich ()`



a) b) c) d) e) f)

Figure 3.6: Starting from acquired pointclouds (a), our algorithm extracts Regular Arrangements of Planes (RAP). The extracted planes are used to reproject the associated points (b), or approximate the input by a set of planar polygons (c) (zoom shown in (d)). The normal distributions (Cartesian projection) of the input cloud (e) are very noisy. In contrast, the extracted RAP have clean normal distributions showing the extracted regularity (f). Circles denote normal locations and their support is indicated by the color of the circles, going from white (few samples) to blue (many samples). Parallel planes are shown with the same color (figures (b)-(d)), and input angles were $\Theta = \{0, \pi/2\}$.

3.5 Evaluation

Datasets. We performed our experiments on various 2D and 3D input scenes. We synthetically generated the input of Figure 3.2 and the Blensor ([51]) laser scan "L-house ". "Stairs" (see Figure 3.7) was acquired using Kinect®. "Nola " (LiDAR) is from [192], "Empire" and "Lans" are courtesy of [87], and "Euler" is from [118]. "Euler-Cut" is a section of the larger scene. Refer to Table 3.1 for scene sizes, and the paper's supplementary material[1] for high-resolution figures.

Scene	#points	#initial	#prims	$\#\chi_i$	#rels	ρ
L-house	5k	67	13	2	61	0.02
Stairs	261k	300	99	2	8281	0.01
Nola	740k	15.8k	847	1	358k	0.004
Empire	1.2M	4.7k	163	17	7025	0.0025
Lans	1.3M	7.4k	490	9	22.1k	0.005
Euler-Cut	586k	4k	965	17	103k	0.004
Euler-Full	3.9M	28.7k	548	5	49.5k	0.002

Table 3.1: Statistics of processed scenes (#points, #initial), and of the RAP retrieved by our method: the number of primitives #prims, representative orientations $\#\chi_i$, and perfect relations discovered amongst primitives in the orientation frames #rels. ρ shows the input feature size given a scene scaled to the unit cube.

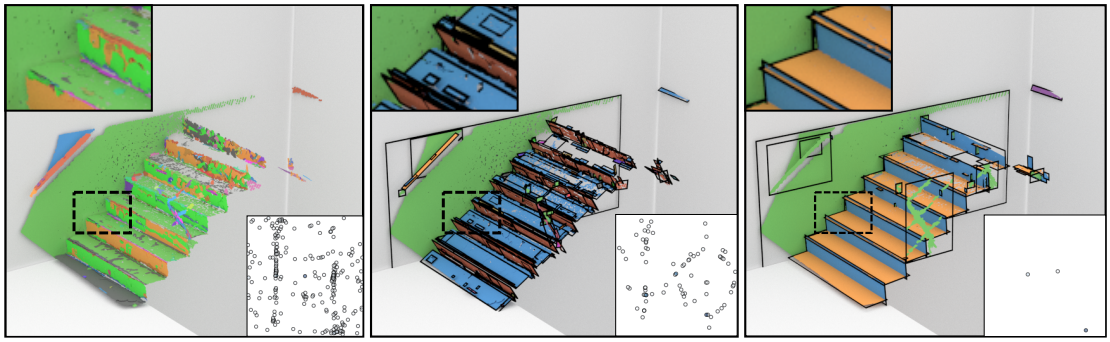
3.5.1 Comparisons

We compared our results to an advanced, RANSAC-based method [142], constrained data fitting GlobFit [90], discrete labeling based PEARL [71], and point set structuring [87].

We used available implementations of the algorithms, except for the latter, where we asked the authors for comparison. RANSAC uses a probabilistic framework that is least sensitive to the notion of finite planes.

We reimplemented the propose, expand, and refit steps of PEARL with finite plane segments using the published α-expansion library. We defined the data points to assign labels to as the 2D or 3D points in the input. The set of labels represent the individual, finite line segments or planar patches. The data term is defined as the distance between the point and the finite primitive. Point to finite planar patch distance is computed as the distance of the point to the minimum area rectangle fitted to the reprojections

[1] `http://geometry.cs.ucl.ac.uk/projects/2015/regular-arrangements-of-planes/paper_docs/MonszpartEtAl_RAPter_SIGG15_SupplementaryMaterial.pdf`



Input segments GlobFit Ours

Figure 3.7: Comparison to [90] on a Kinect® scan. The moderate number of input planes (300) prohibitively increases the likelihood that GlobFit will commit early to an irrelevant relation in the scene yielding over-regularization. There, stair steps were aligned to a single plane spanning the whole staircase, and risers were rotated orthogonally. In contrast, our approach correctly reconstructs the stairs, handlebars, and the wall, initialized with these 300 *or* all 1500 input planes. Input angles $\Theta = \{0, \pi/2\}$.

of the points already assigned to the primitive. The smoothness term is non-zero, when neighbouring points within ρ distance are assigned to different primitives. The complexity cost is defined as a function of the number of primitives with non-zero number of points assigned to them. This formulation based on iteratively expanding finite primitives produced more visually pleasing results than the naïve formulation with infinite primitives, with the added benefit of the parameterization being more easily tunable. PEARL's dense formulation frequently made it necessary to downscale the input point clouds.

Point set structuring takes an arrangement of planes as input, and benefits more from our regularization as preprocessing. Using their full pipeline on our input still yields unregularized outputs (*c.f.,* Figure 3.9, Table 3.2). Except GlobFit, other methods have no notion of global relations, and hence are out-performed very easily by our method.

GlobFit. We used an improved version of the published implementation of [90] to perform extensive experiments on our scenes. We found that: i) GlobFit heavily relies on the quality of relations it first commits to; ii) memory demands quickly rise beyond practical magnitudes. We had to sort the primitives from our initialization in decreasing order w.r.t. assigned numbers of points to achieve any output. In Stairs, Nola, and Lans we observed a degradation of the quality of the output related to the number of input planes. *E.g.* in the Stairs (Figure 3.7) scene, given the 15 largest

planes as input, we both were successful at reconstruction with perfectly oriented planes gathered in a single frame. Using a richer set of 300 planes (of 560) GlobFit had a much lower chance of initially picking the right relations and could not recover from such mistakes. Similar problems arose in Nola ($>$ 120 planes, Figure 3.10), and Lans ($>$ 300 planes, Figure 3.11). Empire is heavily polluted with outliers, which required us to provide GlobFit with the 13 largest planes only (see Figure 3.9). Otherwise, output was a meaningless series of well-distributed, parallel planes.

3.5.2 Results

We evaluated our approach by reconstructing scenes with varying noise levels, sampling, complexity and regularity, both in 2D and 3D. The numbers of input planar segments are reported in Table 3.1, as well as the input scale parameter ρ, the number of output primitives, identified representative orientations (frames), and discovered pairwise perfect relations: $\#rels = \sum_i \binom{\sum_j \chi_{i \to j}}{2}$.

Generally, RAPTER produced more accurate segmentations of the input pointclouds, and critically, arrangements with higher regularity. Most importantly, we simultaneously preserved independent orientations for smaller groups of planes. We reorient points in the input pointcloud using the normals of the assigned planes in our output segmentation. To visualize the distribution of point normals in a scene, we create histogram plots by mapping them to a rectangle using HEALPix's[2] Cartesian mapping. A saturated circle shows high concentration of normals, a white circle denotes less populated, but concentrated orientations, well preserved by our method. Matching colors in the outputs indicate parallel normals ($< 0.01^\circ$).

2D. We prepared a rasterized image to demonstrate vectorization, shown in Figure 3.4. The input was created from an RGB image by thresholding at 40% luminosity. Despite the differences in sampling density and variations in feature scale (thick and thin walls), we obtain a good reconstruction of the main layout of the room, and also manage to recover the spatial layout in ill-sampled locations, while respecting sudden local orientation changes and preserving important details. Also note the significant reduction in the uncertainty of the orientations in the image. The extracted RAP capture the

[2] `http://healpix.sourceforge.net/`

different orientation frames (denoted by same color) across multiple feature scales. Due to PEARL's spatially regularized formulation, it managed to recover the well-sampled, principal directions in the drawing. In contrast, RANSAC used more primitives to explain the hatched areas than the more meaningful borders (see Figure 2.1).

We created the simulated scan of a room in Figure 3.2 to demonstrate the importance of regularization and preservation of details. We used the input angle generators $\Theta^* = \{\pi/3, \pi/2\}$ to solve the scene. Despite the apparent simplicity of the solution, note, that there are four different orientation frames in the scene, therefore the challenge really is to preserve their independence. Although some triangle edges are almost parallel to rectangle edges with overwhelming popularity, we preserve the more accurate data fit due to enough evidence for existence of that orientation and the spatial regularity cost detailed in Equation (3.5). Our method is especially useful, when biased noise and occlusions would distract other methods bound to spatial reasoning. The resulting RAP managed to capture scene diversity without over-committing to any particular orientation, *i.e.,* room walls (red) or dining table (green).

L-house. We evaluated performance on a synthetic scan corrupted by non-uniform, sparse sampling (Figure 3.8). According to our output normal distribution, the six main directions were detected accurately, even small clusters in the corner were correctly oriented. In comparison, the roof planes returned by RANSAC are not perfectly aligned. PEARL performs well on this example due to spatial regularization and reliable data. GlobFit can correctly identify the reliable relations amongst the small number of input planes (91).

Empire. We illustrate how our approach can be used to reconstruct data corrupted by outliers on Empire (Figure 3.9). In addition to the recognised regularity of the building with three main directions, we properly detected small tilted components around edges. Note the cylinder on the top of the building, approximated by planes favoring the global arrangement orientations. RANSAC over-simplified the geometry, because of the excessive presence of outliers. PEARL managed to produce an arrangement of planes with a more compact normal distribution, however the regular structure of the building was lost during the process (see top view). Many details were missed due to the delicate balance between complexity and spatial smoothness. GlobFit did well given the largest 13 planes as input (out of 4693). Given more, it could not deal with the amount of

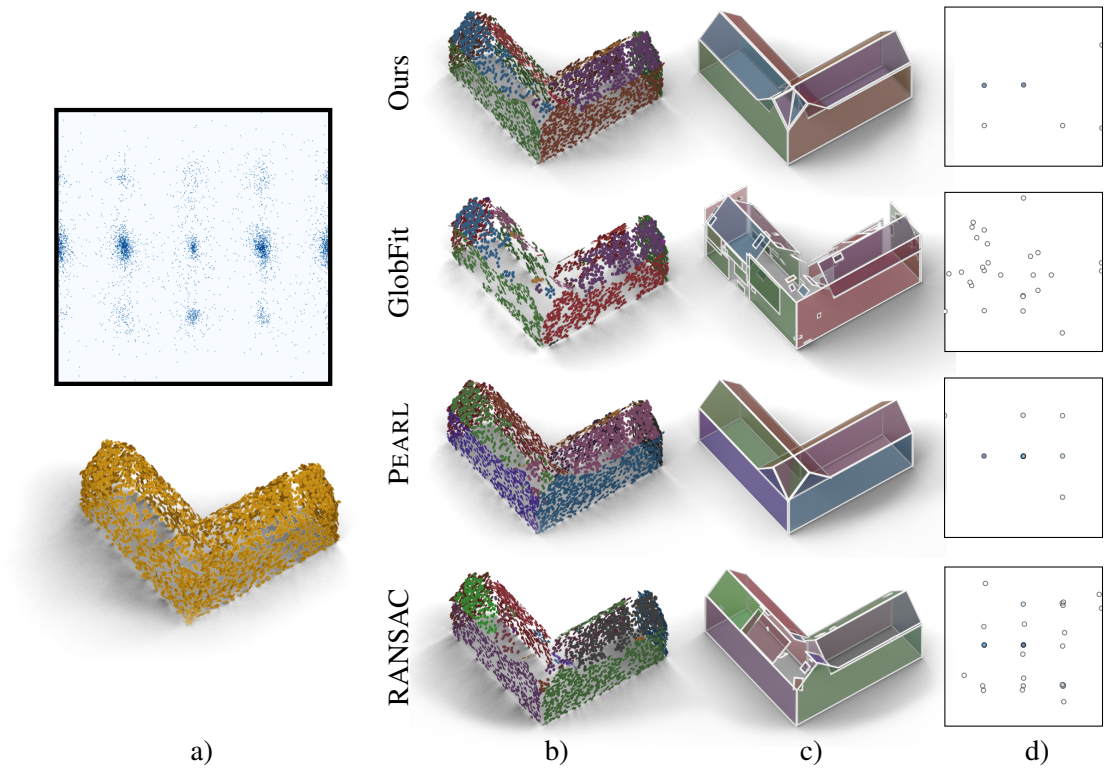


Figure 3.8: Comparison to other methods on "L-house" (a). We show points reprojected to associated planes (b), and approximated by a set of planar polygons (c). Input (a-top) and output (d) normal distributions are mapped to 2D. Input angles $\Theta = \{0, \pi/2\}$.

outliers, hence orientations corresponding to the smaller details at the top of the building are missing from the normal distribution.

Nola. Nola is a good example for our approach (see Figure 3.10). The building is exclusively composed of planar components acquired using a long-range scanner, which lead to large, occluded and ill-sampled regions. The regularity of the geometry is well exploited by our approach, balconies are properly detected and perfectly aligned. Note that all planes are either vertical (walls) or horizontal (floors). RANSAC discovered the main directions, but completely failed with smaller structures and large, slanted planes. PEARL managed to extract several horizontal planes (green balconies in top view) but failed to properly segment the geometry due to data quality. GlobFit could not recover from early commitment to some relations in its input, 120 (out of 15685) most sampled patches.

Lans. In Figure 3.11 our approach faithfully reconstructs various shapes including a cone, an octagonal pyramid, and small non-symmetric tetrahedrons (top of the tower). The relations needed for the cone were not prescribed, hence our approach did not

regularize there. The simultaneous extraction of relations and primitives allowed us to preserve small structures, such as the alcove or the windows in the wall, despite strong variations in sampling. Some relations were recoverable by only relying on the

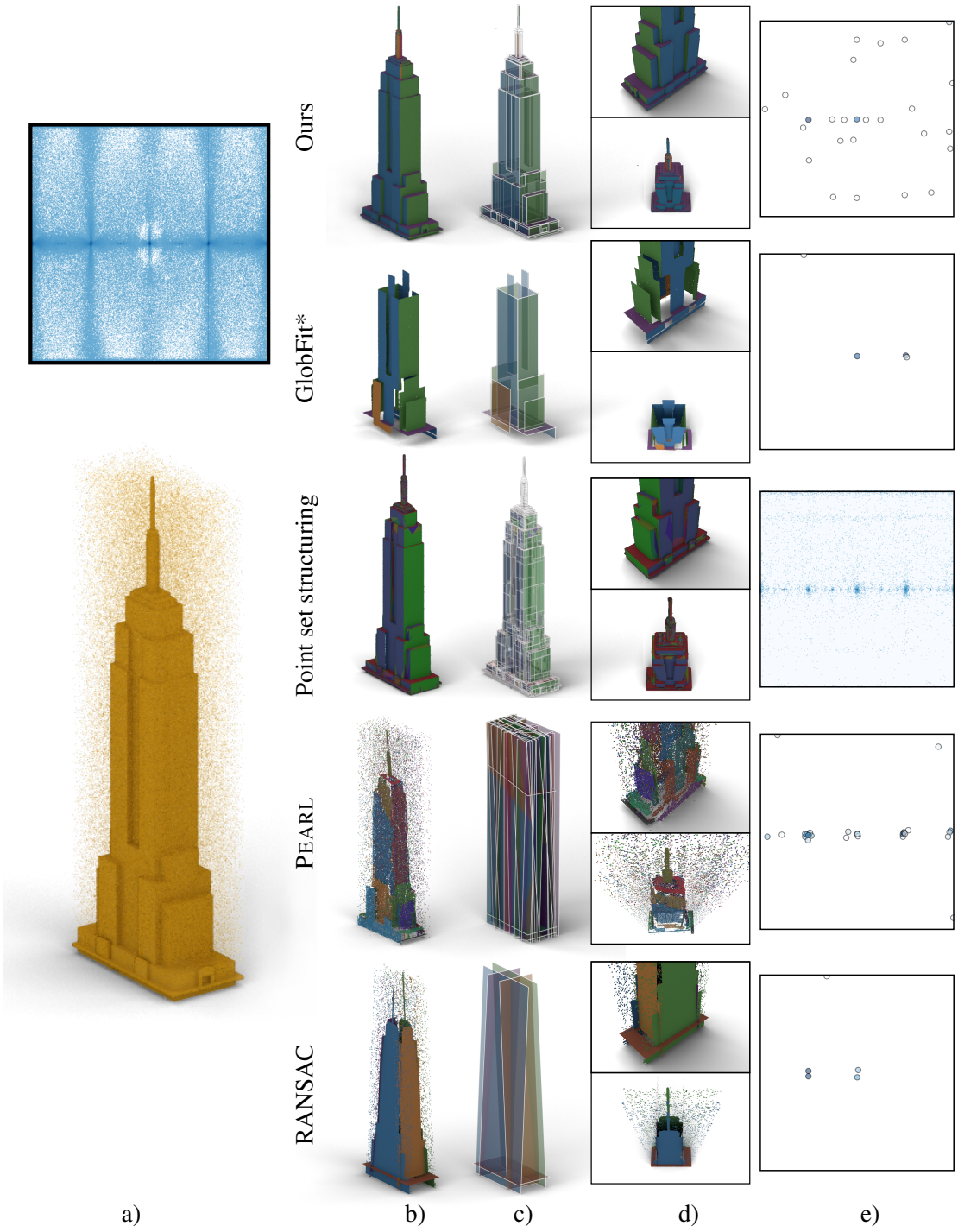


Figure 3.9: Comparison to other methods on "Empire" (a). We show points reprojected to associated planes (b), and approximated by a set of planar polygons (c) (zooms in (d)). Input (a-top) and output (e) normal distributions are mapped to 2D. Main difficulty here is detecting and segmenting structures of very different sizes whilst disregarding outliers. GlobFit only had the 13 (of ~4700) most supported planes as input. Input angles $\Theta = \{0, \pi/2\}$.

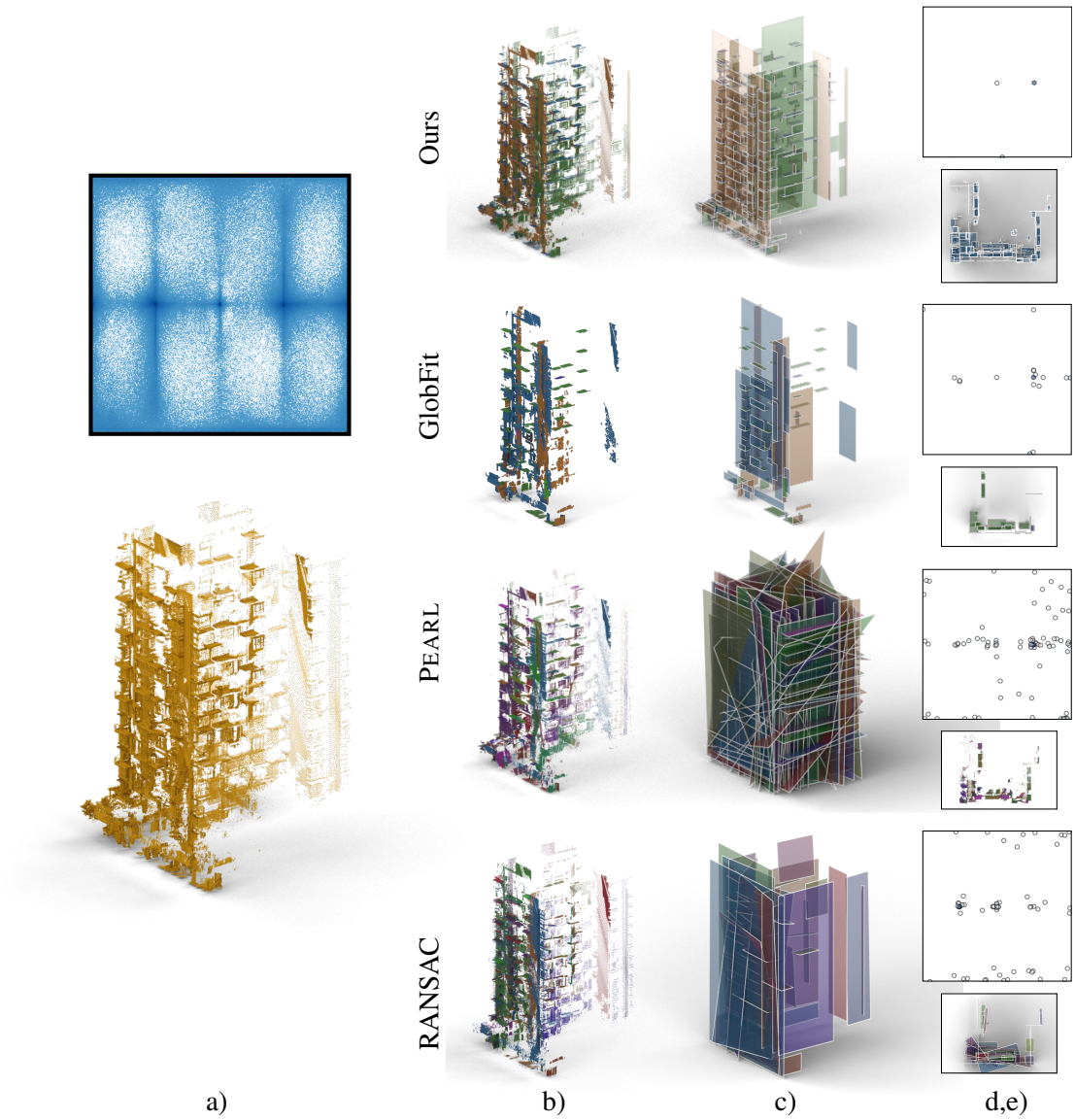


Figure 3.10: Comparison to other methods on "Nola" (a). We show points reprojected to associated planes (b) and approximated by a set of planar polygons (c) (zooms in (d)). Input (a-top) and output (e) normal distributions are mapped to 2D. GlobFit had the largest 120 (of 15865) input planes. Input angles $\Theta = \{0, \pi/2\}$.

data (RANSAC, PEARL). RANSAC failed to differentiate small and nearby similar structures due to the conflict between low complexity and attention to detail (tower wall and roof). PEARL worked with a subsampled pointcloud (10%) due to performance reasons. GlobFit repeatedly over-regularizes the scene by enforcing relations (roof, tower top) even given just a moderate part of our input (out 300 of 7441).

Euler. The power of our multi-scale design is illustrated by the scenes "Euler" and "Euler-Cut" (Figures 3.1 and 3.6), containing large (walls), medium (tables), and small structures (stairs, risers, and table feet). Due to the simultaneous identification of

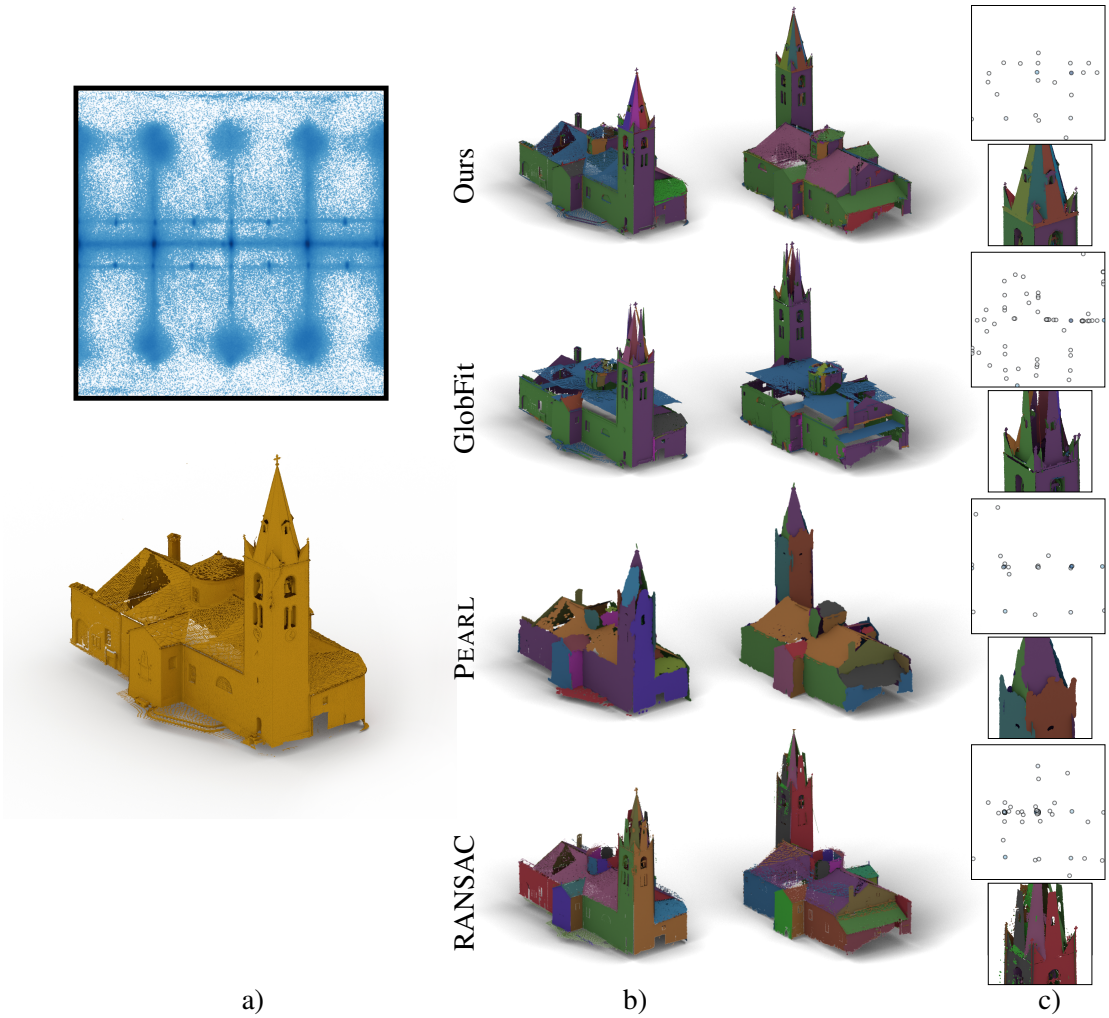


Figure 3.11: Comparison to other methods on "Lans" (a). (b) Points reprojected to associated planes and zooms in (c). Input (a-top) and output (c) normal distributions are mapped to 2D. GlobFit had 300 (of 7441) input planes. Input angles $\Theta = \{0, \pi/2\}$.

primitives and relations, our approach easily outperforms RANSAC when segmenting the geometry. For example, at the stairs landing, where corrupted data is over-merged (by RANSAC), mixing the floor and the three first stairs. Also note the robust repeatability of our approach over cluttered geometry (conference room): where many chairs are segmented similarly (brown front legs, green back legs, seats in dark green, backs two-colored). We could not run other methods (PEARL, GlobFit) on these scenes (500k-4M points).

Generally, well sampled scene parts can allow other methods to recognize some relations. However, RAPTER is robust towards sampling density and low local contrast of features, whilst capable of solving large scale scenes. Most importantly it can correctly identify non-dominant directions besides main scene orientations.

3.5.3 Quantitative evaluation

We evaluated our method quantitatively (Table 3.2) and also show the range of control offered by the regularization parameter (Figure 3.12). In Figure 3.13 we investigated robustness to initialization.

Ground truth. We retrieved ground truth triangle meshes for three scenes, face normals serve as ground truth orientation for points. We only used faces with edges $> \rho$, and ground truth points, where there is a single face within ρ distance due to lack of assignments.

Error metrics. We target to evaluate the successfully identified relationships in the scene through the relative angle of normals assigned to points by different methods (Table 3.2). Specifically, we compare the angle of normals of pairs of points to the angle of corresponding ground truth normals. We report the mean and standard deviation of absolute difference of relative angles: $\forall_{g,h} \quad \left|\angle(\mathbf{n}(p_g),\mathbf{n}(p_h)) - \angle\left(\mathbf{n}(p_g^{GT}),\mathbf{n}(p_h^{GT})\right)\right|$. We considered only points, that were plausibly reconstructed by their assigned planes, *i.e.,* their reprojection was close ($< \rho$) to their reprojection to the ground truth mesh. We were especially interested in how relative relations were successfully recovered *perfectly*, *i.e.,* up to 0.1° compared to the ground truth relative angle of a point pair. In addition to our scalability and ability to respect diversity, numbers show a clear advantage over other methods.

Regularization. We show the amount of control and power the user has through the parameter λ, see Figure 3.12. When large diversity is expected in the input, or the input scans have been preprocessed, one can choose to respect the data and not enforce regularity (left side of the figure). Given prior knowledge about relations in the underlying scene, one can choose not to trust the possibly moderately or very noisy data containing biased sampling and registration mistakes. In most scenarios, the middle column would be the sought outcome. However, especially given registration errors, outcomes towards the right side show the true power of our method.

Robustness to initialization. In Figure 3.13 we evaluated the robustness of the algorithm to the quality of the initialization. Note, that the fragmentation of the initialization depends on the spatial threshold ρ and is only loosely coupled with the threshold parameter τ. We show, how we recover from a wide range of settings and degrade gracefully, when the the parameters were obviously ill-chosen.

Scene (#pts)	Method	Mean (SD)	$< 0.1°$	Recall
L-house (5k)	RANSAC	10.6° (3.6°)	35.1%	**100.0%**
	PEARL	12.4° (3.9°)	34.2%	**100.0%**
	GlobFit	11.5° (3.8°)	73.9%	99.4%
	RAPTER (ours)	**10.1°** (**3.6°**)	**75.4%**	99.8%
Stairs (261k)	RANSAC	**7.7°**(**2.2°**)	13.3%	84.7%
	PEARL	17.6° (4.0°)	16.4%	84.5%
	GlobFit	13.3° (2.9°)	8.1%	**99.6%**
	RAPTER (ours)	9.4° (3.5°)	**65.7%**	98.2%
Empire (1.2M)	RANSAC	**2.8°**(**0.4°**)	20.5%	21.7%
	PEARL	20.8° (4.3°)	9.6%	35.4%
	GlobFit	6.3° (3.0°)	80.5%	65.8%
	[Lafarge&Alliez '13]	4.6°(2.5°)	89.8%	93.7%
	RAPTER (ours)	4.1° (2.0°)	**94.6%**	**99.1%**

Table 3.2: We compare the relative angles of normals of point pairs in the output clouds and normals from ground truth triangle meshes. A point is considered recalled, if its reprojection on its assigned plane is at most at $< \rho$ distance from its reprojection on the ground truth mesh, higher is better. RANSAC achieves a lower mean deviation (lower is better) between the angles of normals by fitting to many well-sampled planes, some false positive primitives fit to outliers. Our high recall scores show that our precision comes with plausibly located output planes. We are especially interested in how well perfect relations were discovered, we therefore estimate precision (higher is better) of identified, perfect pairwise relative relations.

Limitations. Our novel formulation with meta-nodes equipped the algorithm to become robust to the pointcloud size, the number of primitives, and to be able to process large scale scenes as shown in Figure 3.1. However, this comes at the cost of quadratic

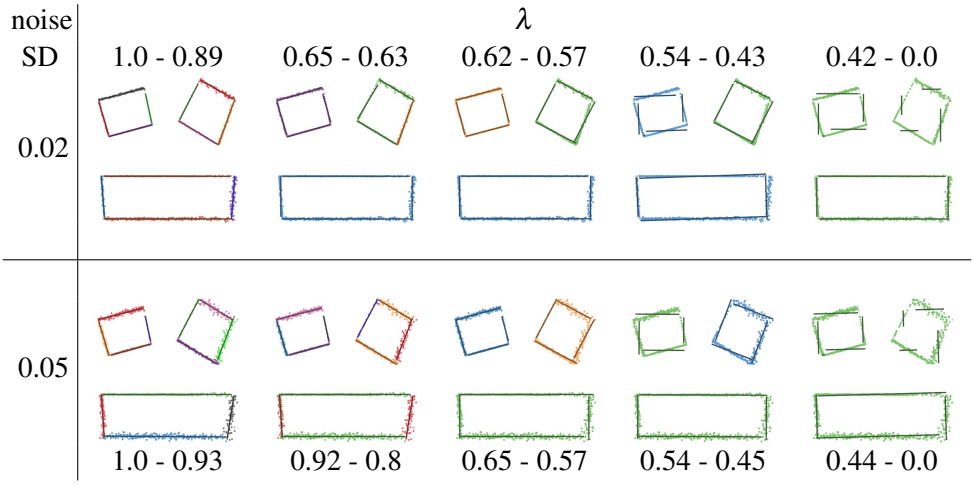


Figure 3.12: Effect of regularization parameter λ given two levels of normally distributed noise. Data was captured from a point scanner in the center of each sub-image, with biased sampling and registration errors. The regularization parameter λ spans between the extremes of allowing a user to either let the data guide the reconstruction, or to enforce a set of orientations on the whole scene. Intervals of λ lead to the same results. For really noisy scenes, higher λ values are appropriate. Input angles $\Theta = \{0, \pi/2\}$.

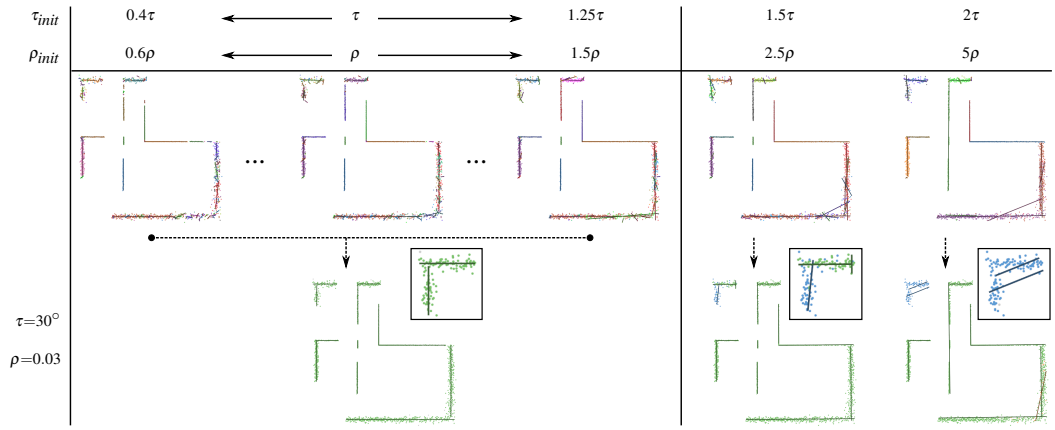


Figure 3.13: Robustness to initialization. Our method is capable of producing consistent outputs given a wide quality variation of the initialization. We performed the initial segmentation using different angular (τ_{init}) and spatial (ρ_{init}) thresholds (top row). We then optimized using $\tau = 30°, \rho = 0.03$ to retrieve the bottom row. Despite challenges presented by erroneously under-segmented point patches, our output performance degrades gracefully. Situations, where the parameters were set too high (top-right) are easily recognized. Hence, one should choose to initialize by conservatively setting the thresholds to lower values. Input angles $\Theta = \{0, \pi/2\}$.

constraints, which requires more computation time during the optimization stage. With our current implementation, runtimes vary from seconds (L-house) to a couple of hours for massive scenes (Euler) on a quad-core Intel i7-4700MQ (2.40GHz, 16GB RAM).

Our solution solves a selection problem iteratively inspired by Isack and Boykov [71], allowing generation of primitives not only from the data but also previously computed approximations. Despite RAPTER's shown efficiency, advancing to very large scale datasets might be difficult. Due to global constraints, concurrent access to a very large database of points is required, which complicates optimisation on distributed systems. However, large scale scenes (*e.g.,* building-scale) can be processed without further optimizations.

3.6 Conclusions

The algorithm in this chapter was designed to abstract raw scans by regular arrangements of primitive planes by *simultaneously* extracting a set of primitives along with their inter-primitive relations. The proposed formulation favors globally consistent RAP, even at the cost of an increased data fitting error. The main novelty in the proposed formulation is to allow less represented orientation groups *not* to be suppressed by more dominant scene orientations. Algorithmically, RAPTER first expands the solution space by generating potential candidate primitives, and then reposes the scan abstraction

problem as an instance of globally coupled primitive selection problem. The resultant algorithm runs from a coarse-to-fine scale, leading to a robust and scalable algorithm as demonstrated on many test scenarios.

An obvious next step is to add support for other primitive types (e.g., cylinders, cones). More interestingly, in the future, we would like to support other relations including equal spacing and equal length in the global formulation. One possibility is to model such second order relations by using pair of primitive pairs (i.e., quartet of primitives) in the candidate generation stage. However, the resultant candidate blowup will require rethinking the optimization formulation without sacrificing the global selection criteria.

Chapter 4

Physical stability as non-visual cue

4.1 Introduction

Acquisition devices for 3D geometry are now ubiquitous. While this has vastly simplified the data gathering process, the raw data still remains difficult to use. A fundamental problem for single view acquisition and even for multi-view consolidated scans is missing observations due to scene occlusion. This problem is particularly acute in busy and cluttered scenes (*e.g.,* indoor environments). This limits how such raw scans can actually be used as existing shape analysis tools fail due to missing information.

Various approaches have been proposed to address the ill-posed problem of *hypothesizing* data in the occluded scene regions. For isolated objects, one can use local context [147, 56] or deform class-specific template shapes to recover the missing parts [121, 8]; for multiple objects, one can employ search-and-classify approaches [111], require multi-view acquisition [80, 100], or interactively capture template models using targeted acquisition [81] or modeling sessions [5]. Such methods, however, assume access to suitable database models, or rely on much more detailed and complete scene data to reliably perform geometric matching.

In this chapter, we use physical stability considerations of static scene objects to hallucinate missing structure information from incomplete 3D acquisitions. Moreover, by considering possible geometry in the unseen part, we can better reason about the visible parts of the scans (*e.g.,* if two objects just touch, or are fused to each other). This is similar to how we as humans regularly extrapolate seen object parts to reason about the unseen scene regions based on the physical plausibility of the resultant object stack, for example see Figure 4.2. Note that by avoiding the surface reconstruction or

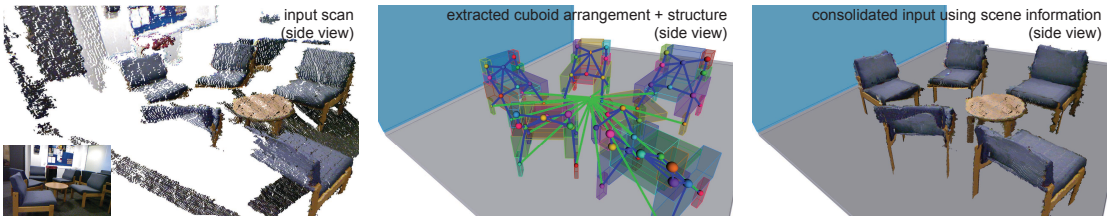


Figure 4.1: Starting from a heavily occluded single view RGBD image (left), we extract a coarse scene structure as an arrangement of cuboids along with their inter-cuboid relations (middle) using physical stability considerations to hypothesize the missing regions/relations. The coarse structure can then be used for scene understanding and manipulation. This result was generated in the automatic mode.

partial shape matching problems, our algorithm does not reply on availability of model templates or predefined priors (see Figure 4.1).

Starting from raw scans, our goal is to abstract indoor scenes as a collection of simple cuboids. We abstract indoor scenes as collections of cuboids as the 'box world' well captures the intra- and inter-object constraints common in man-made environments (*c.f.,* [10, 53, 193]). Creating such an abstraction requires addressing the following: (i) segmenting the scan into groups of points; (ii) fitting a cuboid to each such group; (iii) hypothesizing missing cuboid geometry in the occluded regions; and (iv) determining how the cuboids are mutually arranged and connected to each other. Essentially, our goal is to determine what the cuboids are, what their spatial and geometric parameters are, and how they are relatively arranged (e.g., stacked or fixed to each other) such that they best explain the (incomplete) input data.

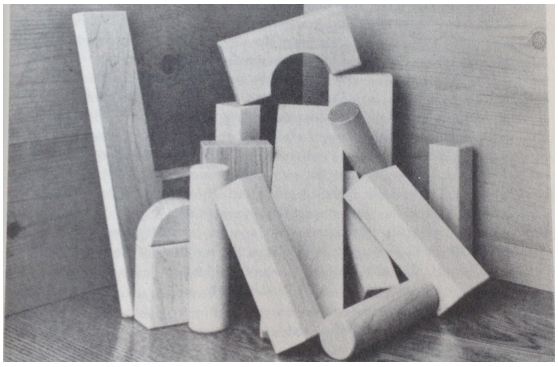


Figure 4.2: Even for scenes with significant occlusion, we as humans, can reason about the actual arrangement (image source: [45]). We develop an algorithm to mimic this based on physical stability of the inferred arrangement (see also Figure 4.15).

The algorithm starts by creating a set of initial cuboids. Based on their pairwise configurations, we extrapolate the initial cuboids to spawn a set of candidate cuboid *extensions* to hypothesize possible completions in the occluded regions. The algorithm then proceeds in two stages: a discrete stage to extract inter-cuboid connections by selecting the cuboid arrangement from a subset of candidate cuboids arrangements that best explains the input; and a continuous stage to refine the cuboid parameters to improve

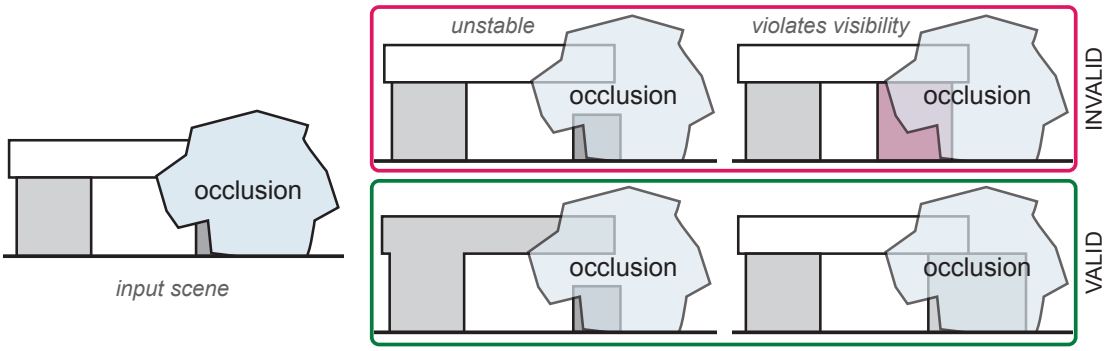


Figure 4.3: Based on incomplete scene observation (left), we can imagine multiple completions. Top-right shows invalid completions: an unstable stack of three cuboids, or a stable stack of three cuboids that violates the visibility constraint. Bottom-right shows valid completions: two fused cuboids and a small cuboid, or a stack of three cuboids consistent with the visibility constraint.

the stability of the current arrangement, while preserving the current cuboid connections to obtain a final cuboid arrangement. Goodness of any cuboid arrangement is assessed based on the physical stability of the configuration and how well the arrangement agrees with the input observation.

We evaluated our algorithm on synthetically scanned scenes with known groundtruth data, on typical indoor scenes obtained from benchmark datasets (about 700+ scenes from the NYU depth dataset with available groundtruth structure data), and on complex office environments with significant occlusions. The tests demonstrate that our algorithm produces high quality structure abstractions of the occluded regions. Moreover, the improvements are significant even in case of scans consolidated using multiple acquisitions (*e.g.,* KinectFusion). Finally, we demonstrate the use of the recovered coarse structure in object retrieval and scene completion.

4.2 Overview

Our goal is to create an arrangement of cuboids as a structural proxy for an incomplete scan of a cluttered scene. By *arrangement of cuboids*, we refer to a set of cuboids with their explicit pairwise contact relations, e.g., two cuboids touch, or are fused to each other, along with geometric attributes for each cuboid.

Starting from the raw data (*e.g.,* single view *e.g.,* data, or KinectFusion data), we first create a set of initial cuboids (see Section 4.3.1). These initial cuboids, however, provide little information about the occluded scene regions. We make the important observation that pairs of such cuboids that intersect when extrapolated into the occluded

regions provide potential geometry hypotheses for the occluded regions (see Figure 4.3). Hence, we generate multiple hypotheses for missing geometry by extending the initial cuboids into the occluded regions. We refer to such extrapolations as *cuboid extensions*. Note that each cuboid can have multiple potential extensions based on its contact types to the other cuboids. In this work, we consider no-contact, fixed contact, and touching as contact type relations. In Section 4.3.2, we describe how we enumerate possible cuboid arrangements $\{A_1, A_2, \dots\}$ where, each arrangement $A_i := \{B_1, B_2, \dots\}$ consists of a set of (extended) cuboids.

Working under the assumption that the scanned scenes consist of static objects, we expect any valid cuboid arrangement to be physically stable, or nearly-stable to account for errors due to coarseness of the initial cuboid estimates. Hence, we measure stability of any candidate cuboid arrangement A_i under the respective inter-cuboid contact types and discard the unstable arrangements (see Figure 4.3). We propose a branch-and-bound algorithm to avoid explicitly checking the exponential set of possible cuboid arrangements.

Finally, we refine the cuboid parameters of the stable arrangement(s) while retaining current contact constraints using a quadratic programming formulation (see Section 4.3.3). We prefer the physically stable cuboid arrangement that requires fewer fixed joints and necessitate minimal volume extension as the simplest solution in terms of geometry that is hypothesized. In the end, we recover a physically valid arrangement of cuboids with both the inter-cuboids interaction types and the respective cuboid parameters.

4.3 Algorithm

Our goal is to infer a physically valid arrangement of cuboids to hypothesize geometry in the missing-data regions in an incomplete 3D point cloud P (e.g., a single view RGB-D image). By physical stability, we require the cuboid arrangement to be globally stable. This is achieved by creating arrangements of cuboids with suitable cuboid geometry and inter-cuboid contact relations (see Figure 4.4).

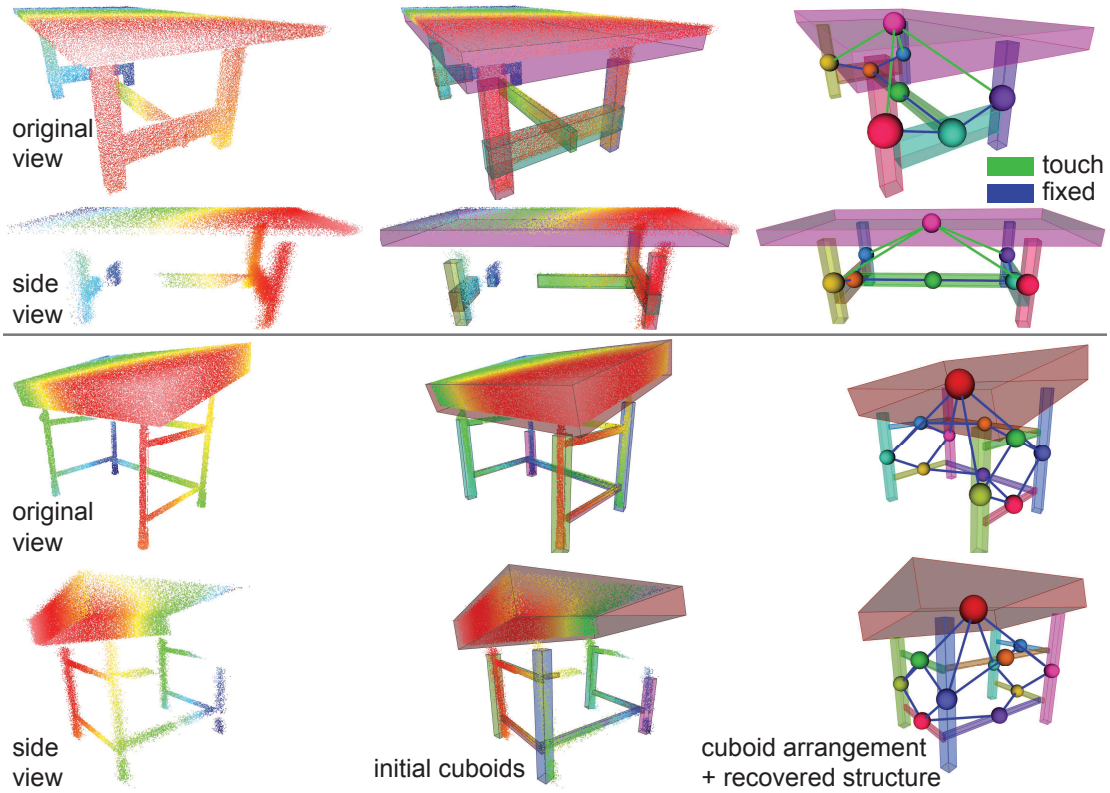


Figure 4.4: Starting from incomplete scans (left), our algorithm starts by creating a set of initial cuboids (middle) that are optimized, both in their geometric parameters and the inter-cuboid contact types to propose a physically-valid arrangement of cuboids (right) that is consistent with the input data.

4.3.1 Generating Initial Cuboids

We detect and remove the points in P associated with the ground and wall planes, group the remaining points into separate clusters of point sets, and create initial cuboids for each such point cluster. For simple scenes, the initialization proceeds automatically, while for more complex scenes, user assistance is required. The generated cuboids are then refined to better align with the image space edges, where available. We now provide further details.

Ground/wall planes. We scale the input point set P to fit inside a unit sphere and align the z-axis to the input up-direction. We use RANSAC to detect the dominant planes [142] in the input scene and sort the planes based on their size (i.e., area of extent). We mark the planar segment with upward normal and lowest z-value as the ground plane, and the other significant (based on an area threshold) planes that are orthogonal to the ground as walls (see [81]). Optionally, the user can mark the ground and wall planes. We remove the points associated with the ground and the wall planes, and group

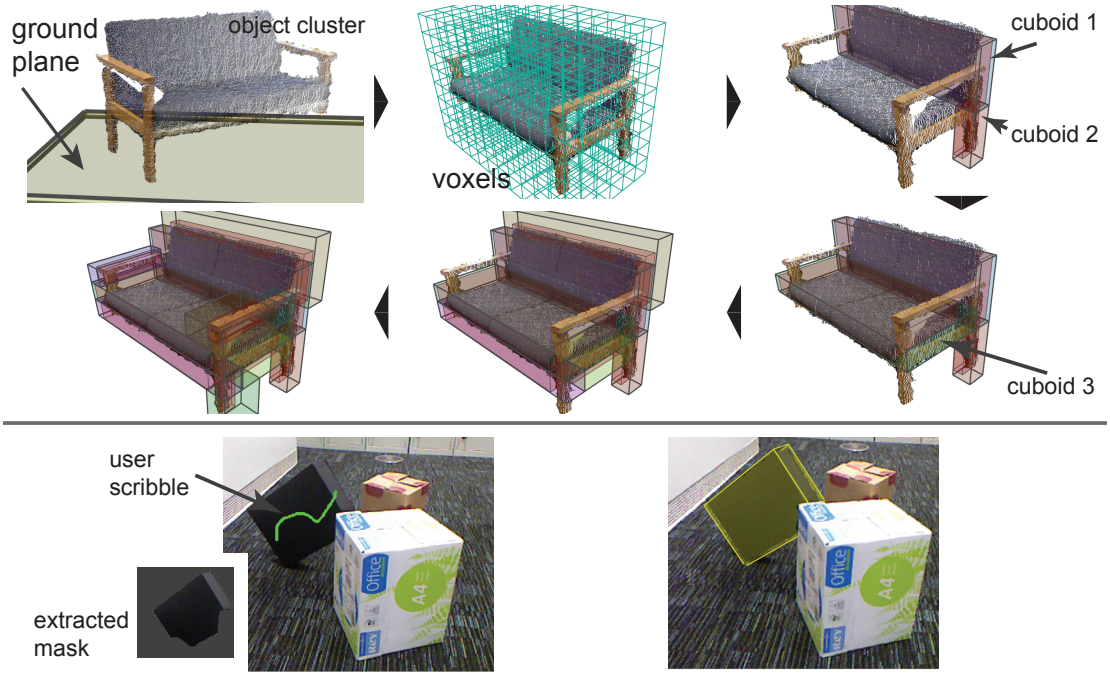


Figure 4.5: (Top) In the automatic mode, cuboids are progressively fitted to an object cluster C_i using a greedy approach. (Bottom) Based on user scribbles, a mask is produced from the RGBD data using depth-augmented grab-cut, and a cuboid is fitted to the corresponding points.

the remaining ones using a connected component analysis (using spatial proximity) as object clusters, say $\{C_1, C_2, \dots\}$. Note that each object cluster C_i denotes a set of points.

Notation. We fit initial cuboids to each object cluster C_i. First we introduce some notations. A cuboid is an oriented box associated with a local coordinate frame. We represent a cuboid B by its center $\mathbf{c}$, a local coordinate frame $\mathbf{F}$, and (s_x, s_y, s_z) as the three size parameters along the three local coordinate axes. Note that along any axis a cuboid can be extended differently in the positive and/or the negative directions.

Automatic cuboid creation. For each object cluster C_i, we fit a bounding box aligned to the ground plane (see Figure 4.5-top). We orient the bounding box based on the best fitting rectangle to the points $p_j \in C_i$ projected to the ground plane. We voxelize (using 0.005 as cell size) and identify the occupied cells. Our goal is to fuse these voxels to form a set of non-overlapping rectangular slabs, each of which is nearly fully occupied. We take a greedy approach by fusing the occupied cells to form rectangular slabs if they contain more than 70% occupied cells (w.r.t. to the respective slab volumes). In the figure, the cuboids are marked as $1, 2, \dots$ based on their order of appearance. In case of multiple candidates we prefer the one with the highest occupancy, and recurse over the remaining cells. In the end we have a set of cuboids for each object cluster.

User interaction. Optionally, the user can scribble strokes on the input image to guide segmentation (see Figure 4.5-bottom). This mode is used in scenes where objects come too close and fail to be separated using connected component analysis. Based on the strokes, we use a depth-augmented version of the original grab-cut segmentation [132] to select a group of points or mask. We run RANSAC on the selected points to generate candidate planes. The largest of the planes is selected as the primary plane, and the second largest plane is made orthogonal to the primary one. We extract the bounding box determined by these orthogonal directions (third direction is the cross product of the two plane normals). In our tests, such interactions were only necessary in regions of clutter and/or with slanted objects.

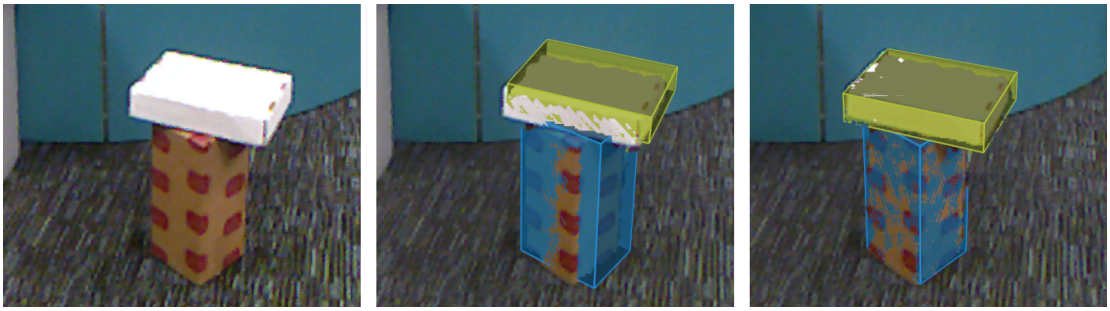


Figure 4.6: The initial cuboids (middle) are adjusted to better align the original image edges in the RGDB data (left) to produce refined cuboid candidates (right).

Image-guided cuboid improvement of initial cuboids. The cuboids generated, either automatically or semi-automatically, are based on the noisy point sets and hence are often misaligned to the image space edges. We refine the cuboids based on available image edge information (see Figure 4.6). Visible corner edges of the cuboids are mapped to the sufficiently close image edges, which are detected using Canny edge-detector (small edges and outliers are discarded). Given a group of cuboids $\{B_1, B_2, \dots\}$, we minimize:

$$\min_{\{B_i, T_i\}} \sum_i \sum_j \|\mathbf{P}(T_i(\mathbf{v}_j^s)) - \mathbf{t}_j\|^2, \tag{4.1}$$

where $\mathbf{P}$ is the known camera projection, $\mathbf{v}_j^s$ is a sampled point on the visible corner edges, $\mathbf{t}_j$ is the closest point of $\mathbf{v}_j^s$ on the corresponding image edge, and T_i is the unknown rigid transformation applied to the cuboid B_i. We update the closest points $\mathbf{t}_j$ at each iteration and use Matlab *fmincon* function to solve the non-linear optimization in

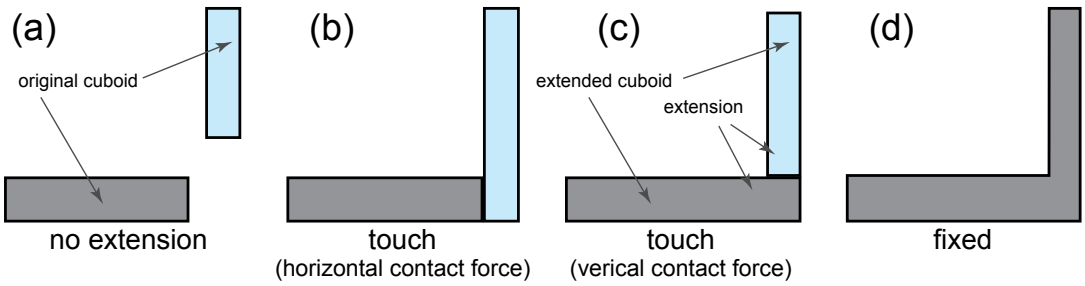


Figure 4.7: 2D Illustration of candidate cuboids generation cases for different contact types. Starting from the initial cuboids (a), we generate different candidate extensions: no extension (a), touching cuboids (b,c), and fused cuboids (d).

Equation (4.1). Note that if the offsets between cuboids and the point cluster degrade after refinement (due to spurious edges), we abandon the refinement.

4.3.2 Inferring Inter-cuboid Contact Types

The initial cuboids do not occupy the occluded regions. In the key stage of the algorithm, we *extend* these cuboids into the occluded regions with the constraint that the arrangement (configuration) of the extended cuboids is physically stable. However, there are many ways of extending the cuboids into the occluded region. We first create candidate extensions and then pose the problem of selecting extensions such that the resultant arrangement of extended cuboids is physically stable. We use the formulation by Whiting et al. [177] to assess physical stability in terms of the (unknown) cuboid dimensions. We now explain the individual steps (see Algorithm 2).

Representing contact types. We encode the discrete interaction among the cuboids as a *contact graph* $G := (V,E)$, where each cuboid B_i becomes a node in V and each pair of cuboids (B_i,B_j) forms multiple candidate edges in E, one for each possible contact type (*e.g.,* touch, fixed).

We make an important observation that the rough geometry of the cuboids is largely determined by *how* they interact with each other and provide good extension candidates for the respective cuboid parameters. Each candidate cuboid pair amounts to possible contacts corresponding to different extensions for no-contact (i.e., disjoint), touching, or fixed (i.e., cuboids are fused together). Each such contact edge implicitly suggests cuboid geometries based on the respective cuboid extensions (see Figure 4.7). Say, any pair of cuboids (B_i,B_j) is extended to be $(B_i^{ij,k},B_j^{ij,k})$, where k indicates the joint type. For example, edge type with no extension ($k = 0$) amounts to retaining the original

Algorithm 2: Inferring Inter-cuboid Contact Types

Input: N initial cuboids $(B_1, ..., B_N)$
Output: N optimized cuboids $(B_1^*, ..., B_N^*)$ and their
contact relations $(J_1^*, ..., J_M^*)$
// Initialization of interaction graph $G := (V, E)$
$G \leftarrow \varnothing$
for $i = 1$ to N **do**
  $V_i \leftarrow B_i$
end for
for each node pair (V_i, V_j) **do**
  // build multi-edges e_{ij}^k
  $e_{ij}^0 \leftarrow (B_i, B_j)$ // edge corresponding to initial geometry of B_i and B_j
  if B_i, B_j can potentially touch **then**
    for $k = 1$ to 3 **do**
      $e_{ij}^k \leftarrow (B_i^{ij,k}, B_j^{ij,k})$ //different extensions as in Figure 4.7
    end for
  end if
end for

// Optimization of cuboid geometry and their relation types
$N_f \leftarrow N^2$ // fixed joint number
$\delta V \leftarrow \pi$ // extended volume
while 1 **do**
  Gather a new combination of potential cuboid extensions $(e_{ij}^k, ..., e_{mn}^l)$
  if no more edge combination **then**
    break
  end if
  Update cuboids $(B_1', ..., B_N')$ based on the extensions on potential edges
  if cuboids penetrate **then**
    continue
  end if
  Calculate current relations $(J_1', ..., J_M')$, count of fixed joints N_f', and volume extension $\delta V'$ after interaction type pruning based on physics stablity
  if unstable according to Equation 4.3 **then**
    continue
  end if
  if $N_f' \leq N_f$ and $\delta V' < \delta V$, **then**
    $N_f \leftarrow N_f'$
    $\delta V \leftarrow \delta V'$
    $(B_1^*, ..., B_N^*) \leftarrow (B_1', ..., B_N')$
    $(J_1^*, ..., J_M*) \leftarrow (J_1', ..., J_M')$
  end if
end while

cuboids, i.e., $B_i^{ij,0} = B_i$. Note that a cuboid B_i can have multiple extensions proposed by different cuboids.

Potential contact-based cuboid extensions. For any candidate pair (B_i, B_j), we extend the cuboids. Without loss of generality, assume B_i is extended to be in contact with B_j. We first determine the extension direction through space partition. The extension length is initialized with the largest corner point to plane distance, *i.e.,* the corner points of B_j to the cube plane of B_i attached to the overlapped half space. Figure 4.7 shows the different extension cases. Note that the type of extension also determines the force bearing faces, when applicable, which is later used during evaluating physical stability.

Optimization formulation. Let $\chi_i^{ij,k} = 1$ denote the corresponding extensions are selected ($\chi_i^{ij,k} = 0$ otherwise), which means, the k-th contact type for the cuboid pair (B_i, B_j). A trivial solution is to select fixed contact types for all the cuboid pair relations to propose a physically stable solution. However, this is overly conservative. Instead, our goal is to propose a physically stable arrangement of (extended) cuboids with minimal number of fixed contacts, i.e., $\min \sum_{i,j,k} \#(\chi_i^{ij,k} = \text{fixed joint})$.

Further, the selected edges should also respect the following conditions: (i) $\chi_i^{ij,k} = \chi_j^{ij,k}$ and (ii) $\sum_k \chi_i^{ij,k} = 1$ for all the edges in $e_{ij}^k \in E$. The first constraint ensures that compatible cuboids are extended, while the second one ensures that only one type of interaction is selected for each cuboid pair relation.

The combination of all possible contact relation candidates is very large. An exhaustive search can take a prohibitive large amount of time to find the optimal solution, which is not acceptable in interactive applications. First, we prune the possible pairs of cuboids if their extensions violate the visibility constraint, *i.e.,* if the extended proxies fall in the visible regions as informed by the source depth data. Then, we prune extensions using a branch-and-bound approach as described next.

Pruning the solution space. We prune contact relation types based on an important observation: If a candidate cuboid arrangement is unstable even with some of its contact relations marked as fixed, it cannot be made stable by changing the marked fixed type relations to touching or no-contact. Therefore, starting with root node where all the interaction types are set to be fixed, we expand the node as a search tree by changing relations at one edge from fixed to other types. If an expansion results in an unstable arrangement, the whole subtree is pruned to save unnecessary computation (see Figure 4.9).

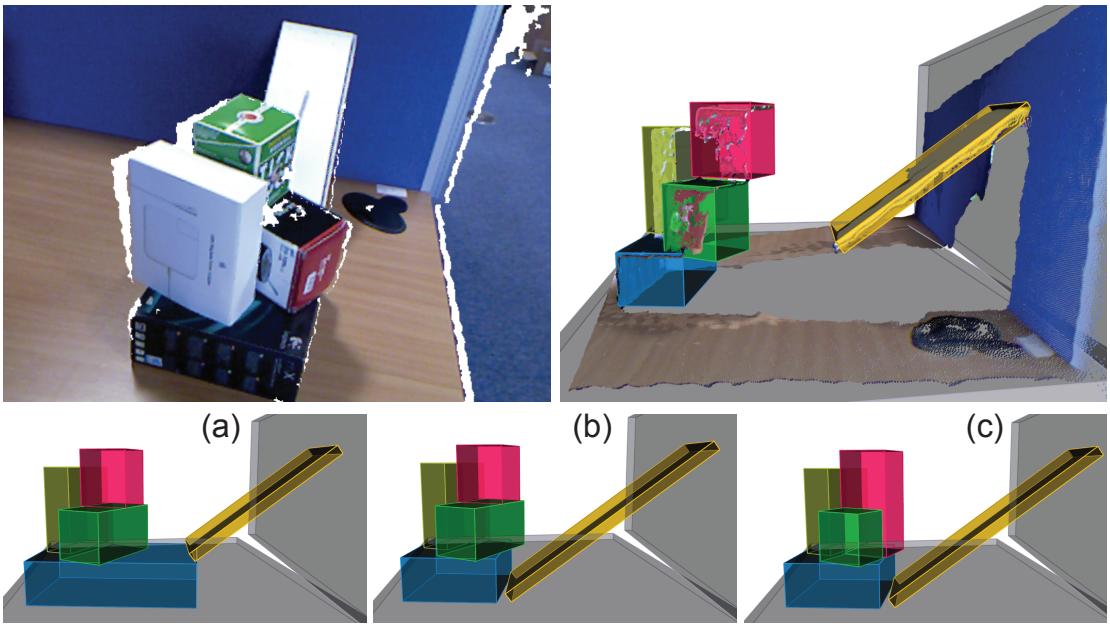


Figure 4.8: A single view RGBD scan (top-left) can result in a physically unrealistic arrangement of initial cuboids (top-right), which can have multiple explanations in terms of arrangements of extended cuboids (bottom). Our formulation favors the one with the fewest number of fixed joints; and the smallest overall extension volume (c).

Assessing global stability. Note that we cannot assess physical validity simply based on local reasoning. For example, in Figure 2.2a, all the pairs of touching cuboids by themselves are physically unstable, but the arrangement as a whole is still stable; while in Figure 2.2b, it is the other way round. Such scenes were common in the complex scenarios scanned and analyzed in Section 4.4.3. See also, Figure 4.14. Hence, in order

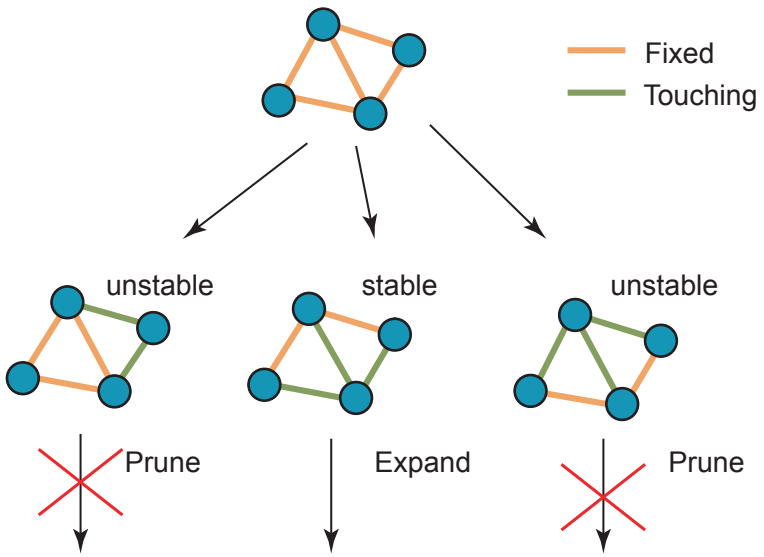


Figure 4.9: Branch-and-bound approach to prune infeasible contact relation type assignments.

to evenly quantify stability for such arrangements, we have to simultaneously consider the effects of all the cuboids.

Assessing physical stability. The stability of a cuboid arrangement is judged through its static equilibrium. This amounts to the set of contact forces between the cuboids to balance the arrangement under self load. Similar to [177], we treat each cuboid as a rigid block, and position a contact force at each contact point on the interface. The number of contact points on the interface can vary according to the contact cases (1 for point contact, 2 for edge contact, and 4 for face contact). The contact forces are then parameterized at each interface face between two touching cuboids as the normal force $\mathbf{f}_n$, and the friction forces $\mathbf{f}_u$ and $\mathbf{f}_v$ (aligned to the normalized face directions $\mathbf{e}_n, \mathbf{e}_u, \mathbf{e}_v$, respectively). Since the interface forces are encoded in the local coordinate system, only 3 scalars are required in the final computation to capture their respective magnitudes. If two cuboids are fused to each other, we treat them as a single rigid body and ignore their contact forces.

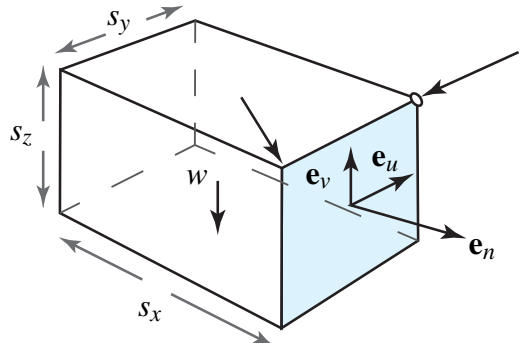


Figure 4.10: For physical stability an arrangement of cuboids should satisfy zero force and zero torque conditions. The condition can be expressed in terms of the (unknown) forces at the interface faces and the weight w of each cuboid expressed in terms of its (unknown) dimensions. In this figure, we highlight one interface face in blue. Note that the cuboid weight at its center of mass is given by $w = \rho s_x s_y s_z$ for a fixed density ρ.

In static equilibrium, the net force and torque acting on the assembly should be zero. Gathering the constraints at each cuboid yields a linear system of the form,

$$\mathbf{D}\mathbf{f} + \mathbf{w} = 0, \tag{4.2}$$

where $\mathbf{w}$ captures the self weights of the cuboids (no external force is considered), $\mathbf{D}$ captures the coefficients of the force/torque equilibrium equations, and $\mathbf{f}$ captures the stack of (unknown) forces at the interface faces (see Figure 4.10).

In order for a cuboid arrangement $\mathbf{A}$ to be physically stable, each i-th interface face forces $\mathbf{f}_n^i$ should be non-zero to act as compression force, and $\mathbf{f}_u^i, \mathbf{f}_v^i$ should satisfy the static friction law (c.f., Section 3 in [177]). Note that the cuboid weights are parameterized in terms of the cuboid dimensions. In order to account for inaccuracies, we seek a least norm solution to Equation (4.2) as:

$$\mathbf{E}_s(\mathbf{A}) := \min_{\mathbf{f}} \|\mathbf{D}\mathbf{f} + \mathbf{w}\|^2 \; s.t. \; \mathbf{f}_n^i \geq 0 \quad \text{and} \quad |\mathbf{f}_u^i|, |\mathbf{f}_v^i| < \mu \mathbf{f}_n^i, \tag{4.3}$$

where $\mathbf{f}_n^i \geq 0$ indicates the compression force constraint, and μ is the static friction coefficient, default 0.7 in our system. If the minimized energy $\mathbf{E}_s$ of Equation (4.3) is below a certain threshold (10^{-3} in our tests), we deem the cuboid arrangement with the assigned interaction type information to be potentially stable, and further refine the cuboid dimensions as described next.

4.3.3 Refining Cuboid Parameters under Contact Constraints

Finally, we refine the cuboid parameters to improve the stability score of the arrangement $\mathbf{A}$, while preserving the connectivity information encoded by the contact graph extracted in the Section 4.3.2. We improve the current stability score using gradient descent. We numerically estimate gradient of the energy with respect to each current cuboid B_i, and greedily select the direction with the maximum norm, *i.e.,* $\text{argmax}_i \|\nabla \mathbf{E}_s(\mathbf{A})|_{\gamma_{B_i}}\|$ evaluated at the current configuration γ_{B_i} of the cuboids B_i. Note that we only vary parameters for one cuboid at a time. We take a small step along the negative of the normalized direction with all the refinements for the other cuboid parameters set to zero. We denote this full configuration vector as $\Gamma_{\mathbf{A}}$. We project the solution to preserve the active interactions as specified by the corresponding contact graph. Specifically,

$$\alpha^* := \underset{\alpha}{\text{argmin}} \|\mathbf{E}_s(\mathbf{A} - \alpha\Gamma_{\mathbf{A}})\|^2 \text{ s.t. } f_i(X) = 0 \;\; \forall e_i \in \tilde{E}, \tag{4.4}$$

where $\tilde{E}$ denotes the *active* edge set in the current contact graph G, and f_i denotes the contact constraints (e.g., the corresponding cuboids touch) as indicated by the corresponding contact type thus resulting in a QP formulation. For example, if two cuboids touch, we add a constraint to ensure that the corresponding interface faces remain touching. At the end of each iteration, we update the cuboid parameters as

$\mathbf{A} \leftarrow \mathbf{A} - \alpha^* \Gamma_{\mathbf{A}}$. We refine the parameters for each plausibly stable arrangement. In case of multiple solutions having same number of fixed contacts, we pick the one with the smallest overall extension volume. For example, in Figure 4.8 the bottom-right solution is selected.

4.4 Evaluation and Results

We processed synthetic and real scenes from recorded and public datasets in order to measure the quality of the optimized proxy approximation. Both the connectivity and geometry of their arrangements were evaluated (when suitable groundtruth was available). We calculated precision, recall, and f-measure for the connectivity validation, and used an L_1 norm based metric to evaluate the improvement of the proxy dimensions.

Datasets: We tested our algorithm on four categories of scenes: (i) synthetic scan simulations allowed us to evaluate robustness against sampling variations, initialization quality, and increasing amount of occlusion due to changing view-points; (ii) 700+ scenes (with available groundtruth) were evaluated from the NYU2 dataset [151]; (iii) a variety of indoor scenes with different levels of occlusion and complexity were recorded to evaluate retrieval capabilities in increasingly occluded environments; and (iv) KinectFusion type scenes were recorded to show the increasing importance of our solution when the scene complexity increases. We also assessed the suitability of the recovered cuboid-based structures for scene completion and manipulation tasks (see Section 4.5).

Evaluation metrics and ground truth. We evaluated how well we can recover the structure information in the scenes, and how much this improves the approximation accuracy of the recorded objects. The structure information was encoded in a support graph similar to [151]. We compared the initial and optimized structure graph to a ground truth graph. Our optimized graph, as described in Algorithm 2, contains edges between two proxies labelled “touch” and “fixed.” The ground truth graph was created manually (or automatically in case of scenes from NYU2) to encode the correct relationships using these labels. Walls, floors, ceilings were assumed to be objects with zero mass. Hanging relationships were labelled “fixed.” The initial graph was assumed to contain solely “touching” edges. Finally, we calculated initial and optimized precision-recall

		INITIAL		OPTIMIZED	
TYPE	#Scenes	PR (%)	F_1	PR (%)	F_1
Synthetic	12	42.1 / 26.7	32.2	96.2 / 84.7	87.2
NYU2	700	34.7 / 47.3	40.0	55.3 / 70.8	60.5
Recorded	20	70.1 / 49.5	56.5	95.7 / 96.6	95.9
Kinfu	2	37.2 / 28.2	32.1	96.4 / 84.0	89.7

Table 4.1: Quality evaluation for structure graphs.

(PR) ratios, and converted them to f-measure ($F_1 = \frac{200PR}{P+R}$). A summary of our results can be found in Table 4.1.

For those scenes where we have access in the real world (to perform measurements by removing occlusion), we evaluated the accuracy gain of the proxy approximation. We compared the L_1 norms of the proxy extents compared to the ground truth both in the initial and optimized scenes (Equation (4.6)). The threshold $\varepsilon = 5\%$ was used to filter changes that could have been induced by sensor noise. The error improvements were summed weighted by the ground truth proxies' proportion in the scene as:

$$e_i = \frac{|s_i - s_i^{(GT)}|}{s_i^{(GT)}} \; ; \; c_i = \begin{cases} s_i^{(GT)}, & \text{if } |e_i^{init} - e_i^{optimized}| > \varepsilon. \\ 0, & \text{otherwise.} \end{cases} \tag{4.5}$$

$$Error(scene_k) = \frac{\sum_{j=1}^{\#proxies} \sum_{i=1}^{3} c_i \, |e_i^{init} - e_i^{optimized}|}{\sum_{i=1}^{3j} c_i}. \tag{4.6}$$

To collect the ground truth, we imagined the smallest bounding proxy of the real world objects and physically measured the three side lengths using measuring tape. We denoted these $s_i^{(GT)}$ and compared them to the measured side lengths in each setting ($init$, $optimized$), denoted by s_i in Equation (4.5).

4.4.1 Synthetic Data

We modeled scenes (chair, stacking boxes, table with chairs) to evaluate the stability of the solution regarding the quality of initialization, arrangement complexity and change of viewpoint. The models were then virtually scanned using Kim et al. [81] and then initial cuboids estimated as described in Section 4.3.1. We evaluated the quality of the structure graph compared to the ground truth using F_1 score (best being 100%).

(i) Robustness to initialization. In order to evaluate the robustness of our approach towards the quality of initialization, we synthetically perturbed the parameters of the initial cuboids (within $\pm 5\%$ of original size, and within 15° of original orientation).

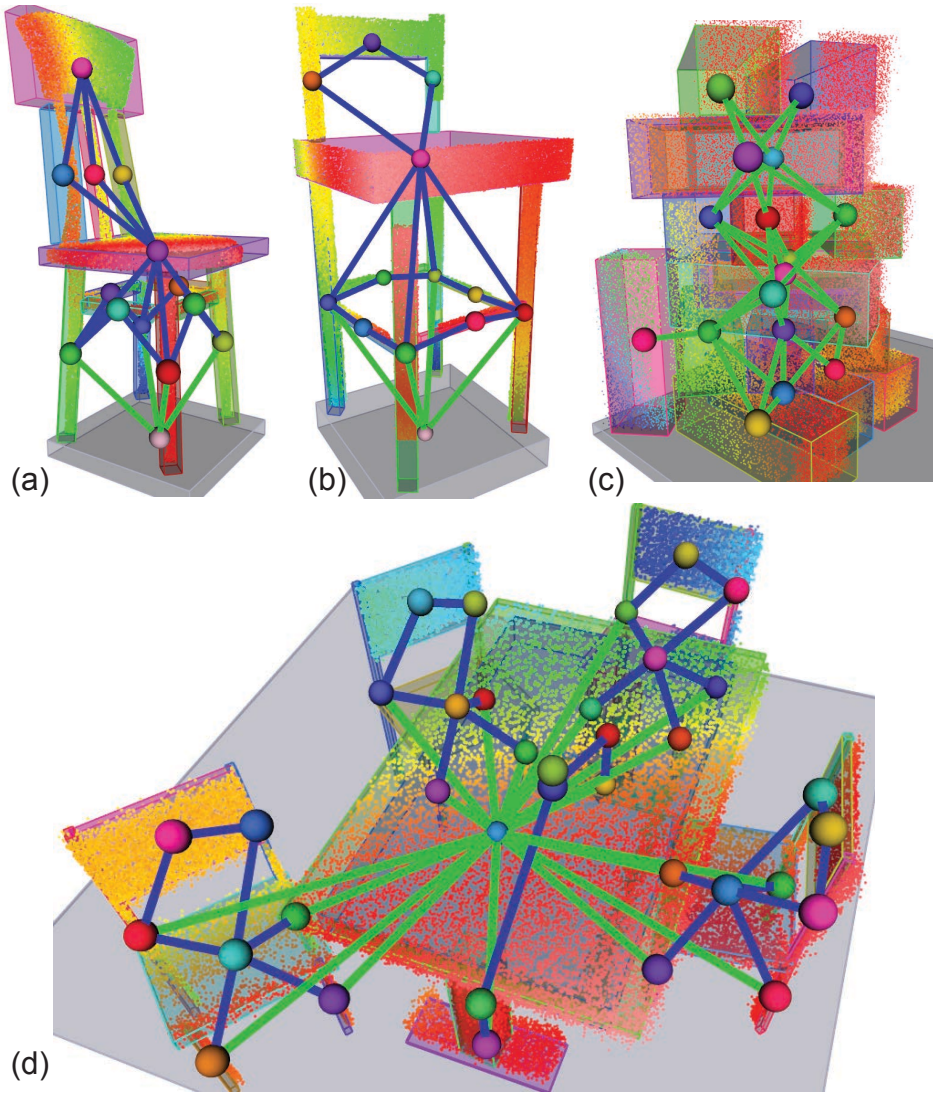


Figure 4.11: Synthetic scenes created and sampled to evaluate stability. Robustness to perturbation of initialization (a,b), arrangement complexity (c) and change of view-point (d) were measured.

We discard any perturbation if it results in intersecting cuboids. We then used such arrangements as input to our core algorithm and got consistent results with respect to ground truth structure graphs over several runs (about 20) for each of the bookshelf and chair scenes (Table 4.2).

We also tested the robustness towards random perturbation of proxy densities. We changed the densities of up to 20% of the cuboids selected randomly, each up to $\pm$5x. We found the desks and sofas to be more fragile allowing 2-3x changes, while other scenes (aligned box stacks) to be more robust (5-7x variations) over 20 runs. Note that we switched off image-based refinement for these experiments as due to the perturbation the recovered proxies might not be containing their generating points.

(ii) Robustness to change of viewpoint. We evaluated the effect of varying occlusion due to different view-points by re-sampling the top-right scene in Figure 4.11 from different views. The most significant challenge was due to the lack of an initial proxy when no corresponding point samples was recorded. This occurs when an object part is entirely occluded in the scene. Some errors are inherited from the decaying sensor accuracy when targeting surfaces from skew view-angles.

In total we simulated 5 different viewpoints, the best F_1 score improvement was $25.5 \rightarrow 90$. Our method has two solutions to resolve inconsistencies. If enough redundancies were present, some proxies (legs of chairs and tables) can be hallucinated based on symmetry. In other cases an existing proxy is extended to make the scene stable (Figure 4.8). This minimally resulted in one false positive and two false negative edges in the structure graph.

(iii) Robustness to arrangement complexity. We evaluated how well our method handles densely connected structure graphs. Synthetic scenes with 6, 9 and 18 stacking boxes were scanned and optimized. Additionally the connectivity of real recordings (Figures 4.13 and 4.15) were also compared in Table 4.3. The 90+ F_1 scores show that our method can successfully handle both densely connected structure graphs (median valence 5 in Figure 4.12) and large number of proxies (56 in Figure 4.13).

4.4.2 Performance on NYU2 Dataset

We tested our algorithm on a wide range of scenes (700+) from the NYU2 dataset. For initialization, we split each depth image into components based on the available segmentation masks, and fitted initial cuboids as described in Section 4.3.1 aligned to the ground plane. A large fraction of the scenes contain simple relationship hierarchies, such as object-on-floor or object-on-object-on-floor. A small ratio of these objects have incorrect structure due to occlusion. Stacked objects, when present, were vertically aligned, making the cuboid-based completion rather simple. We left out scenes with transparent objects.

scene	0°	5°	10°	15°
chair_01	20.0→100	20.51→100	20.51→97.8	17.14→87.8
chair_02	21.43→100	20.69→100	14.29→97.5	14.29→94.7

Table 4.2: F_1 scores (initial $\rightarrow$ optimized) of optimization after perturbation of initialization. The initialized proxies were randomly rotated around their centroids.

	Fig. 4.12	Fig. 4.15	Fig. 4.13
# proxies	18	17	56
# contacts	41	29	64
max valence	6	11	11
median valence	5	2	2
average valence	4.55	3.35	2.32
min valence	1	1	1
$PR(\%)$ init.	85.19 / 56.10	58.82 / 34.48	15.56 / 10.77
$PR(\%)$ optim.	90.24 / 90.24	100.0 / 89.66	98.33 / 90.77
F_1 init.	67.65	43.48	12.73
F_1 optim.	90.24	94.55	94.40

Table 4.3: Robustness to arrangement complexity.

Ground truth structure graph. We evaluated the correctness of our scene analysis by comparing the optimized scene graphs to the ground truth support graphs using updated versions of [52]. These graphs are manually created, and contain semantic information (some proxies are much larger then their pointclouds, since a certain type of furniture and its size is assumed). Some of the proxies come with the labels "floor", "wall", or "ceiling". Further, several scenes have multiple floor regions (from NYU2) but only one floor proxy. This produces ambiguous correspondence. For the remaining scenes we extracted the support graph from the NYU2 dataset and matched it to the ground truth proxies. We used the resulting support graphs as ground truth.

Automatic evaluation. The ground truth graphs automatically retrievable with the method above contained 40%-85% of the edges a manual annotator would insert, influencing the scores reported in Table 4.1. Our algorithm was designed to resolve

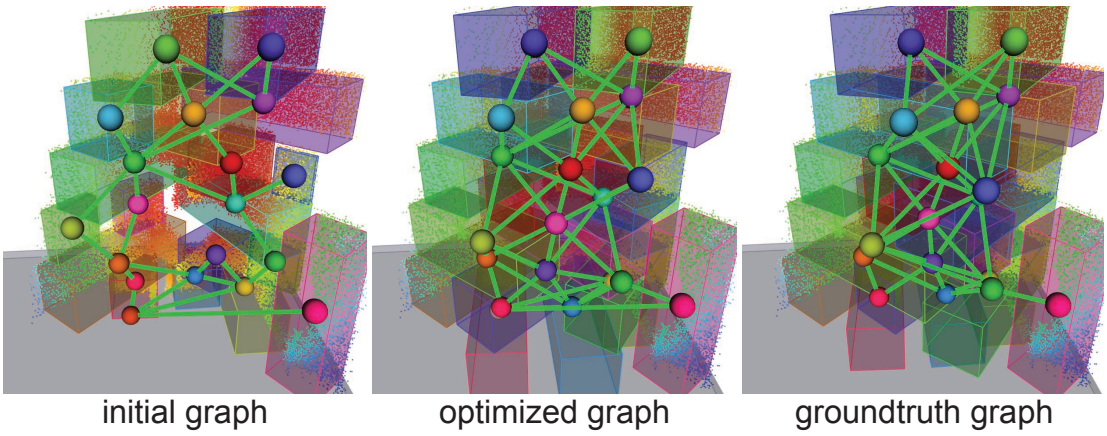


Figure 4.12: Robustness to arrangement complexity. A scene with 18 proxies was reconstructed with F_1 score 90.24. The structure graph has 41 edges, median valence is 5. Some of the edges are missed because of the minimal extension principle in areas of occlusion.

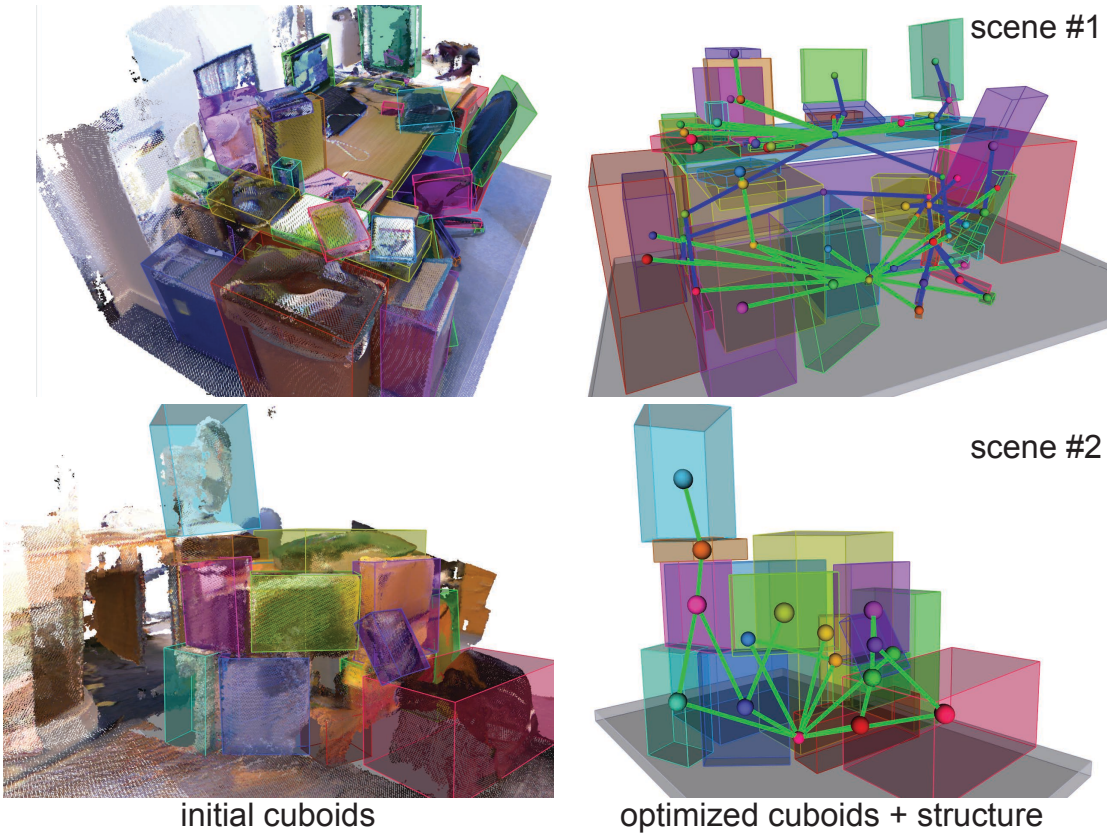


Figure 4.13: Initial and optimized cuboids with extracted structure for two multi-view recordings of cluttered scenes (see the paper's supplementary material for details).

proxy intersections, which assumes a constant approximation scale and no "inclusive" relationships. We pruned those "touching" edges both from the initial and optimized structure graphs before evaluation, resulting in higher precision but lower recall scores. (All the results are provided as part of the project's supplementary material[1] for visual inspection.)

4.4.3 Cluttered Indoor Scenes

We obtained single view Microsoft Kinect® scans of several (20+) test scenes with increasing complexity. These scenes were focused on situations with larger amount of missing information compared to the NYU2 dataset. Due to occlusion, recovering the structure graph and geometry posed a significantly higher challenge. The objects in these scenes are often locally unstable, but globally stable requiring a global analysis. For example, see the multiple cuboids in case of the sofa objects. Recorded scenes contained: vertically stacked box arrangements, slanted boxes, non-cuboid objects,

[1] `http://geometry.cs.ucl.ac.uk/projects/2014/imagining-the-unseen/paper_docs/imaginingTheUnseen_supplementary_compressed.pdf`

single/multiple pieces of office furniture; and scenes with significant clutter and cyclic support relationships (Figure 4.15). The scenes had 2...23 (average 8) objects and 2...29 (average 10) touching and fixed contacts. In addition to estimating the validity of the structure graphs (Table 4.1) we evaluated the correctness of the geometric information recovered by physically examining the scenes and comparing them to the optimized outputs. The comparison showed that our method improved the proxy approximation from 65% to 97% averaged over all the recorded scenes of varying complexity. This means that where our method made significant ($> 5\%$) changes to the scene geometry, we reduced the approximation error by 32% on average (see Figure 4.16 and the paper's supplementary[2] materials for details).

4.4.4 Multi-view Recordings

In order to investigate, whether the addressed problems of occlusion can be resolved by easily accessible 3D reconstruction systems, we captured 2 scenes ($\sim$1000 RGBD frames each) from as many viewpoints as accessible. We used PCL's Kinfu LargeScale implementation. The experiments show that even when such an involved reconstruction system is at hand, the main problems persist. Parts of a convoluted scene can be occluded from every view-point or it can be hard, impractical or impossible to make a recording from view-points that would reveal the necessary structure details of the scene. Creating the ground truth support graph for a scene containing 50+ objects takes $\sim$1 hour by

[2] http://geometry.cs.ucl.ac.uk/projects/2014/imagining-the-unseen/paper_docs/imaginingTheUnseen_supplementary_compressed.pdf

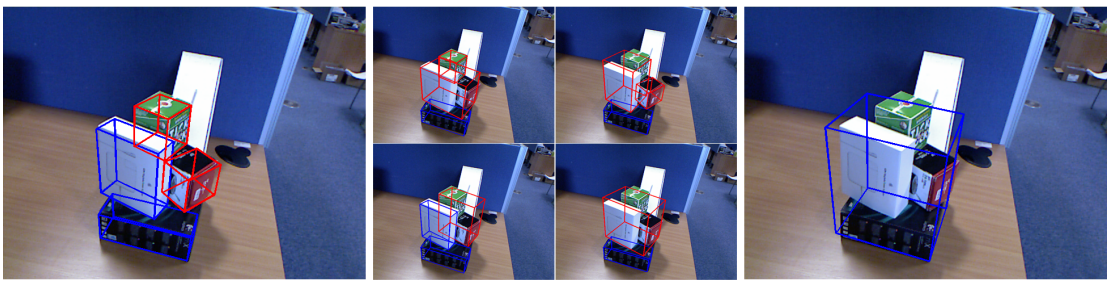


Figure 4.14: Output of Jia et al. [73] on a scene with severe occlusion. Stable cuboids are shown in blue and unstable ones are shown in red. Given the ground truth segmentation, initial boxes are fitted after adjusting their orientation based on the supporting relations (left). In order to reach the physical stability, the algorithm merges neighboring segments during the optimization (middle). The optimized result is to merge all the regions to be stable (right).

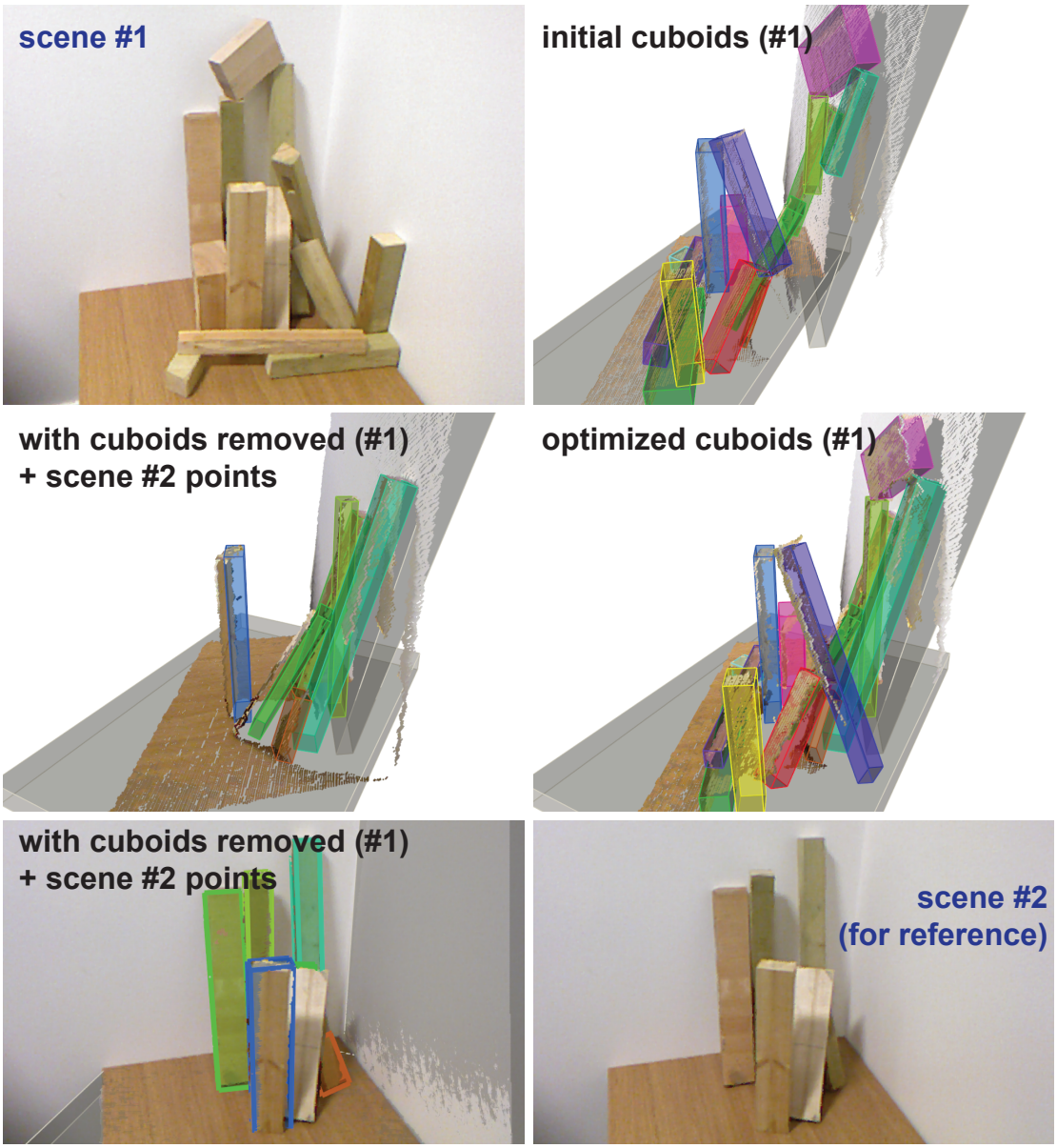


Figure 4.15: Initial cuboids obtained from scene #1 are optimized to produce a physically stable cuboid arrangement. To show clearly how well the optimized boxes are aligned to groundtruth, we progressively remove some cuboids from the optimized cuboid arrangement (virtually) and compare with groundtruth obtained by physically removing the corresponding wooden blocks from the real world scene (scene #2). Last row shows how well the hypothesized cuboids follow the groundtruth in occluded parts.

disassembling the scene for physical measurements. Our algorithm converged in 20 seconds up to 98.3% precision and 90.8% recall (F_1 : 94.4) (Figure 4.13).

4.4.5 Comparison

In a related approach, Jia et al. [73] proposed a 3D volumetric reasoning algorithm for parsing RGBD images with 3D block units. At a high level, we both use stability considerations, but with different goals: we for *reasoning about unseen parts*, and they

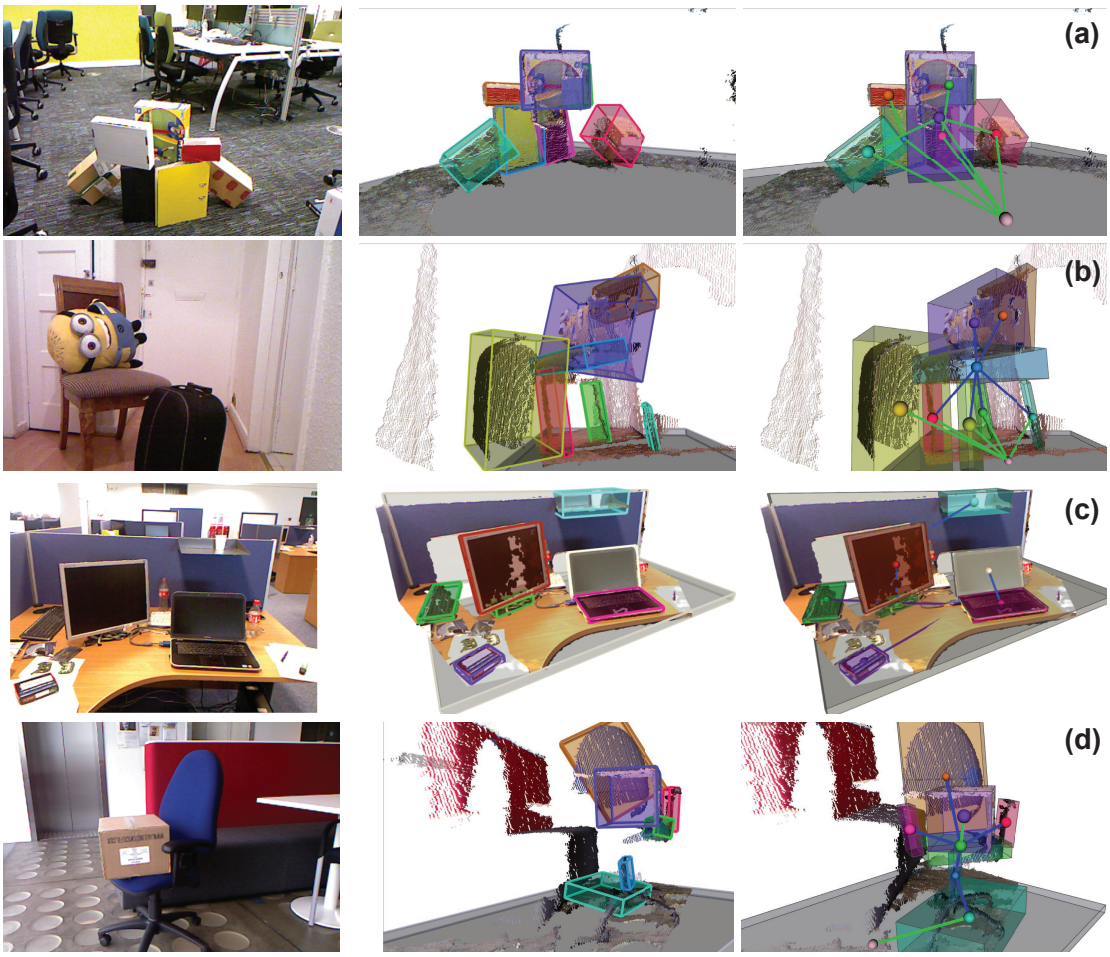


Figure 4.16: Algorithm results on various scenes with occlusion and stacking. Ground planes (or desktop) are shown in gray. Scribbles were used for the slanted objects; rest of the algorithm runs automatically. Left-to-right: Input RGBD scan; initial cuboids and optimized cuboids along with extracted contact graphs (back view).

for image segmentation and interaction (support and stability). Hence, under severe occlusions as for heavily occluded parts, their algorithm does not perform very well. For example, see Figure 4.14. One interesting future direction is to take their output as an initialization to our system and hypothesize the missing parts of the scene.

4.4.6 Performance

User interaction. While for simple scenes automatic cuboid generation is sufficient, for more complex scenes the user may be required to delete spurious boxes or annotate scribbles to guide the initial cuboid generation. When necessary, user interaction took 10-30 secs, and proved to be simpler than annotating in 3D.

Timing. We tested our algorithm on single view RGBD images captured by a Microsoft Kinect camera. Once the initial cuboids were generated, the algorithm took

10-20 seconds to converge depending on the scene complexity (for scenes with < 30 cuboids). All the results are included in the related paper's supplementary material[3].

4.5 Applications

(i) Structure-guided completion. Once we have good-quality scan of a particular type object and its complete cuboid structure in a single view RGBD image, the structure-guided completion can help transfer the scan guided by its structure to complete the missing information of partial scans of same type objects.

The completion is done by registering the good-quality scan to partial scans assisted by the cuboid structure. Let $\mathbf{A}^1 = \{B_i^1, i = 1, \ldots n\}$ denote the cuboid structure of the good-quality scan and $\mathbf{A}^2 = \{B_j^2, j = 1, \ldots m\}$ the extracted cuboids for the partial scan. We established a correspondence between the two cuboid arrangements using a spectral matching approach proposed in [89] by comparing cuboid dimensions in part-level comparisons. Assuming all the objects are on the ground plane, we prune the edge pairs when their height and length differences are larger than 0.05m and 0.1m, respectively. Afterwards, for two candidate corresponding cube edges, we project them to the ground plane and solve for the aligning rigid transform by assuming the correspondence of the projected end points of the two edges. The computed transformation is then refined via ICP using the 3D points. Figure 4.1-right shows an example of scan completion using the sofa scan along with its extracted cuboids-based structure.

(ii) Model retrieval. Completed cuboid structures can also be used to retrieve high-quality 3D models for the corresponding low-quality point clouds. Instead of learning a mapping between local depth patches and 3D model labels [145], we directly employ the completed structure as a global context to assist shape matching. The model with highest matching score is selected as the most similar model. We first get an initial alignment between repository models, retrieved using object keyword search, and the target structure by aligning their upright orientation [149]. We assume repository models to be upright oriented. For the cuboid structure, the up direction is the same as the ground normal, and the right direction is selected from any horizontal edges of a randomly picked cuboid. The model is then translated and scaled to fit into the bounding

[3] `http://geometry.cs.ucl.ac.uk/projects/2014/imagining-the-unseen/paper_docs/imaginingTheUnseen_supplementary_compressed.pdf`

box of the structure. Finally we sample rotation angles about the up axis of the scene to roughly estimate an alignment. Matching score is estimated based on model vertices that are covered by the cuboid structure. Specifically, matching score is defined as the sum of the matching ratio of covered vertices and the uniformly sampled points inside the cuboid structure, and the matching ratio of covered vertices and all the model vertices. We refine the alignment between the point clouds and the 3D model using ICP, and use the model with the best alignment score. Figure 4.17-right shows three retrieval results.

(iii) Image manipulation. RGB images corresponding to the original depth scans can be edited using the abstracted structure (*i.e.,* ground, wall planes and physically stable cuboid arrangement). In each case, we created a background image by identifying and deleting image pixels corresponding to the points in P (in 3D) that fall inside or are close to the completed cuboids. We ray-cast the retrieved object models, which are interactively re-positioned (by moving on the ground plane), to create the foreground layer including the shadow map on the ground plane (default light from above). Holes in the background layer were completed using PatchMatch and composited with the rendered foreground layer. Note that the original camera view is used for such edits. Figure 4.17-bottom row shows few examples.

4.6 Conclusion

I presented an algorithm to discover the coarse structure from partial scans of cluttered scenes with significant occlusion. The structure is captured in the form of abstracted cuboids and how they are mutually arranged. Starting from an initial set of cuboids, semi-automatically created, the algorithm proposes several possible extensions into the occluded regions and selects the one that is physically stable and requires minimal extension. The framework was evaluated on scenarios involving cluttered scenes with stacked and piled objects, and the recovered structure was shown to be useful for applications including scan completion and scene understanding.

Limitations and future work. Due to the fixed density assumption, the algorithm can fail when this assumption is violated by large imbalance among object densities. One possibility would be to assign different densities based on image-based attribute classification. A natural extension of this work would be to consider other types of primitives (*e.g.,* cylinders, spheres). Finally, the output of our algorithm being



Figure 4.17: Based on single view RGBD scans (left), our algorithm extracts a set of stable arrangement of cuboids (middle). The arrangement is then used to retrieve matching 3D meshes with cuboid decomposition and used for scene modeling (right). (Bottom row) Retrieved models can be repositioned, rendered, and combined with the original RGB to synthesize novel manipulated images.

physically realistic can directly be integrated with physics-aware image manipulation and simulation systems resulting in interesting and non-trivial mixes of real and virtual objects and their interactions.

Chapter 5

Accurate dynamics as non-visual cue

5.1 Introduction

Collisions capture suspense, build anticipation, and pack drama. Naturally, they remain an integral part of movies, games, and entertainment. Creating a good real-world collision sequence involving multiple objects, however, is difficult. While the act of smashing two objects into each other so that they collide in a certain way is already non-trivial, the setup quickly becomes unmanageable when additional colliding objects are to be collided, or adjustments are required to the object trajectories. Such changes can easily require many further iterations and recordings, and become a tedious trial-and-error process. Moreover, trying out multiple collision iterations with expensive or fragile objects may not even be a realistic option.

Accurately capturing real-world collision sequences poses further challenges. On one hand, such sequences necessitate high to very high framerate capture, thus making state-of-the-art methods such as KinectFusion, unsuitable candidates for 3D acquisition. On the other hand, even high-framerate video data only provides partial 2D information, both in space and in time (see Figure 5.2). A fundamental problem arises due to unavoidable occlusions near collision instances, which prevents direct observation of the actual collision processes in any acquisition setup.

While the physics of object collisions is a challenging problem in itself, well-developed high-level models exist to reduce its complexity. One widely used assumption is that of infinite object stiffness, *i.e.,* ideally rigid motions. Whereas such rigid body simulations are widely used in games and movies, the task of setting up a collision with the right initial conditions remains tough: many parameters, such as velocities,

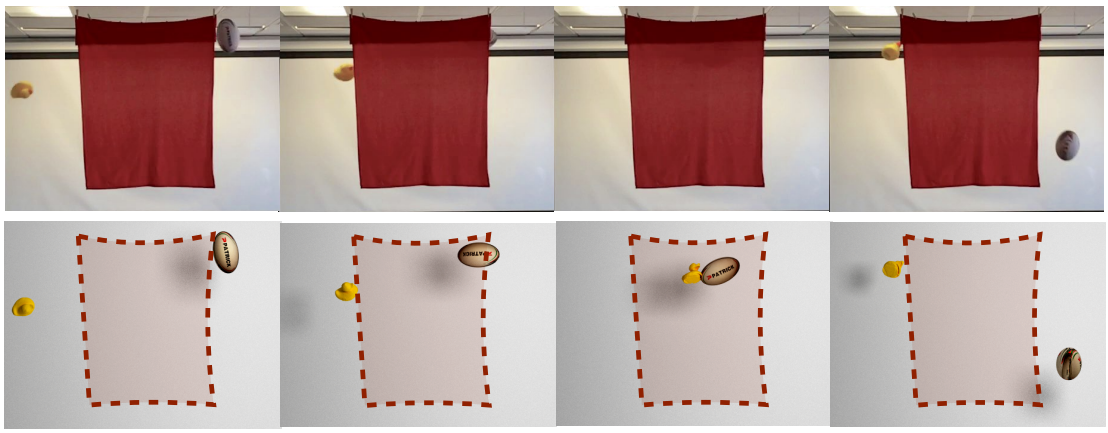


Figure 5.1: Starting from an input video of a collision sequence behind a curtain (top), SMASH reconstructs an accurate physically valid collision (bottom) using laws of rigid body physics for regularization. Note the reconstructed spin (*i.e.,* , angular velocity) of the objects.

mass ratios, and coefficient of restitution, have to be correctly guessed and adjusted. Given the nonlinear nature of the underlying physics, such a rigging up of a desirable sequence is problematic and typically requires extensive prior experience. Further, physical parameters (*e.g.,* coefficient of restitution) may not be readily available for the object pair at hand. Finally, such an approach only enables forward simulations.

In this work, we propose to marry the benefits of the above setups. The user records collisions between pairs of objects simply using a high framerate video (a smartphone in our setup). The video data, however, lacks depth information and is noisy. We first automatically extract a set of candidate 3D positions from the video, and semi-automatically initialize a sparse set of orientations. We then formulate an optimization using regularization derived from conservation laws of rigid body collisions to reconstruct space-time trajectories of the participating objects. This step utilizes the original 3D models of the colliding objects, assumed to be captured beforehand. As output, we can directly *read off* physical collision parameters from the reconstruction that can readily be incorporated into existing physics engines to recreate and reauthor modifications of the recorded collisions. For example, in Figure 5.1, we show the replay of the reconstructed collision sequence happening behind the curtain. Note that we obtain high-quality reconstruction results even when the most relevant section (in time) of the original collision remains fully occluded in the input video.

We evaluate our method both on a range of synthetic evaluation setups and complex real-world collision examples, and generate new collision sequences using an easy

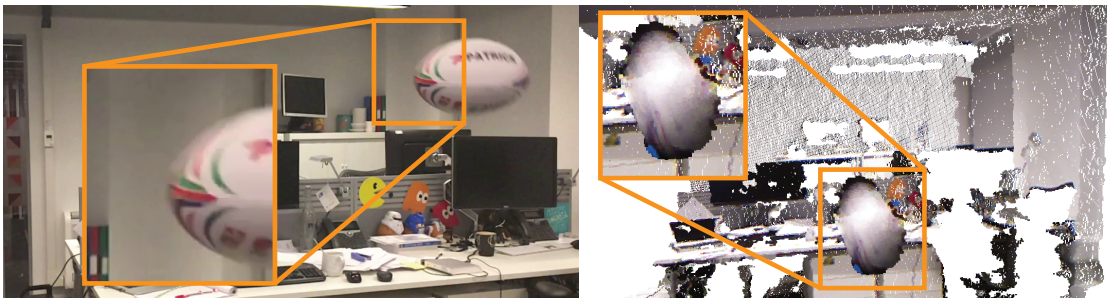


Figure 5.2: Accurate acquisition of collision sequences is difficult as objects move fast and are typically occluded around collision time: a video captured by a smartphone (left) showing motion blur, and an RGBD scan using Kinect (right) that is noisy and partial.

authoring workflow. In summary, we propose an algorithm to reconstruct physically valid collisions from an input video, and perform collision analysis without access to exact object geometry. The estimated collision parameters can then be used to stage new collisions sequences.

5.2 Formulation

Our goal is to acquire pairwise relevant parameters of object collisions simply based on an input video and reconstruct a physically valid motion. As output we produce a space-time recording of the collision event that can be re-used in a variety of ways: to set up new collisions that behave faithfully to the original recording, to introduce new objects, or even to stage complex interactions between objects by combining multiple recordings.

We first briefly review rigid body dynamics (Section 5.2.1) as the governing equations provide important building blocks for our optimization. We then formulate the problem (Section 5.2.2) and identify the appropriate conservation laws to constrain solutions to the space of physically realistic ones (Sections 5.2.3 and 5.2.6). We solve the problem in an energy-minimization framework using the above components. In the following, we use bold lower-case letters for 3D and 4D vectors (*e.g.,* $\mathbf{p}$), and reserve bold upper case letters for matrices (*e.g.,* $\mathbf{R}$), see Table 5.1. All our measurements are in world space units.

5.2.1 Rigid Bodies

By using a rigid body model we represent an object by its motion around its center of mass $\mathbf{p}$. In addition to $\mathbf{p}$, each body has an orientation, represented as a unit quaternion

$\mathbf{q}$ (we denote the equivalent rotation matrix by $\mathbf{R_q}$). Due to rigidity, we only need to consider linear and angular velocity ($\mathbf{v}$ and ω, respectively). A body further has a mass and a moment of inertia (calculated for a reference orientation), denoted by m and $\mathbf{I}_0$, respectively. While $\mathbf{p}, \mathbf{q}, \mathbf{v}$, and ω change over time, m and $\mathbf{I}_0$ are assumed to be constant. For notational simplicity, we keep the time dependence, *e.g.,* $\mathbf{p}(t)$, as much as possible implicit and simply write $\mathbf{p}$. The instantaneous velocity of any point $\mathbf{x}$ on a rotating body is given by $\mathbf{v} + \omega \times (\mathbf{x} - \mathbf{p})$.

Frictionless collisions between two objects are typically modelled with a scalar impulse j acting along a collision normal $\mathbf{n}$. For two bodies a and b, we denote their respective variables with a superscript, *e.g.,* $\mathbf{p}^{\mathrm{a}}$ is the center of mass of object a. We will use ^{c} for variables related to the collision event. The impulse changes pre-collision velocities into a set of post-collision velocities (denoted with superscript ^{pre} and ^{post}, respectively) such that the scalar relative velocity $v_r = \mathbf{v}_r \cdot \mathbf{n}$ at the point of collision satisfies $v_r^{\mathrm{post}} / v_r^{\mathrm{pre}} = -c$. Here, c is the coefficient of restitution, which is related to the amount of energy that is transferred into a reversal of the object's motion. The remainder is *lost* for the simulation and dissipated into heat, sound, or work to deform the internal structure of the objects.

In rigid body solvers we typically use a chosen value for c to compute j, and with it the post-collision velocities. The impulse acts anti-symmetrically to conserve momentum, both linear:

$$\begin{aligned} \mathbf{v}^{\mathrm{post,a}} &= \mathbf{v}^{\mathrm{pre,a}} + j\mathbf{n}/m^{\mathrm{a}} \ , \\ \mathbf{v}^{\mathrm{post,b}} &= \mathbf{v}^{\mathrm{pre,b}} - j\mathbf{n}/m^{\mathrm{b}} \ , \end{aligned} \tag{5.1}$$

and angular, where the instantaneous change of the angular momentum $\mathbf{k} = \mathbf{I}\omega$ is

$$\begin{aligned} \mathbf{k}^{\mathrm{post,a}} &= \mathbf{k}^{\mathrm{pre,a}} + ((\mathbf{x}^{\mathrm{c}} - \mathbf{p}^{\mathrm{a}}) \times j\mathbf{n}) \ , \\ \mathbf{k}^{\mathrm{post,b}} &= \mathbf{k}^{\mathrm{pre,b}} - ((\mathbf{x}^{\mathrm{c}} - \mathbf{p}^{\mathrm{b}}) \times j\mathbf{n}) \ . \end{aligned} \tag{5.2}$$

	inputs	*for each ...*
g	gravitational acceleration	global
f_{fps}	video frame rate	recording
$\mathbf{P}$	camera projection matrix	recording
$\mathbf{q}^{\text{obs,a}}, \mathbf{q}^{\text{obs,b}}$	orientation inputs	annotation
	derived quantities	*for each ...*
$\mathbf{p}^{\text{a}}, \mathbf{p}^{\text{b}}$	object position	cont. time t
$\mathbf{q}^{\text{a}}, \mathbf{q}^{\text{b}}$	object orientation	cont. time t
$\mathbf{v}^{\text{a}}, \mathbf{v}^{\text{b}}$	linear velocity	cont. time t
$\omega^{\text{a}}, \omega^{\text{b}}$	angular velocity (world)	cont. time t
$\mathbf{I}_0^{\text{a}}, \mathbf{I}_0^{\text{b}}$	inertia tensor (local)	object
$\mathbf{p}^{\text{2d,a}}, \mathbf{p}^{\text{2d,a}}$	image-space position	frame (discr. time)
s^{2d}	bounding circle diameter	frame (discr. time)
$d^{\text{a}}, d^{\text{b}}$	approximate depth	frame (discr. time)
	unknowns	*(number)*
β_x, β_{y1}	gravity rotation angles	recording (2)
$\mathbf{b}_3^{\text{a}}, \mathbf{b}_3^{\text{b}}$	parabola translation (collision pos.)	object (2×3)
$\mathbf{q}^{\text{c,a}}, \mathbf{q}^{\text{c,b}}$	orientation at collision time	object (2×4)
$\mathbf{k}^{\text{pre,a}}, \mathbf{k}^{\text{pre,b}}, \mathbf{k}^{\text{post,a}}, \mathbf{k}^{\text{post,b}}$	angular momentum	segment (4×3)
$b_1^{\text{pre,a}}, b_1^{\text{pre,b}}, b_1^{\text{post,a}}, b_1^{\text{post,b}}$	parabola linear coefficients (x coord.)	segment (4×1)
$b_2^{\text{pre,a}}, b_2^{\text{pre,b}}, b_2^{\text{post,a}}, b_2^{\text{post,b}}$	parabola linear coefficients (y coord.)	segment (4×1)
$\beta_{y0}^{\text{pre,a}}, \beta_{y0}^{\text{pre,b}}, \beta_{y0}^{\text{post,a}}, \beta_{y0}^{\text{post,b}}$	parabola rotation angles	segment (4×1)
$\mathbf{x}^{\text{c}}$	collision point (world)	collision (1x3)
$j\mathbf{n}$	impulse times collision normal	collision (1x3)
t^{c}	collision time	collision (1)
c	coefficient of restitution	collision (1)
$m^{\text{b,a}}$	mass ratio	collision (1)

Table 5.1: Notation table.

5.2.2 Problem Statement

In contrast to the typical forward simulation above, our goal is to retrieve the physical parameters directly from a real collision of two objects. This would normally involve a large amount of tedious manual work or complicated capturing setups. We now explain our *inverse approach* to retrieve the physical parameters based on observations of the objects' trajectories. It should be pointed out here that computing absolute quantities (*e.g.,* mass) is not possible without access to a reference measurement. We will not be able to compute an absolute position of the objects on earth from a single video input, and, correspondingly, we cannot compute their absolute masses. However, we can retrieve relative quantities, *i.e.,* the relative positions, and the relative mass of the

objects. We neglect all other external effects (*e.g.,* aerodynamic drag, friction) at work except for a known gravitational acceleration with magnitude $g = 9.81 m/s^2$.

Interestingly, the actual shape of the objects does not play a direct role. Only a related quantity, the distribution of mass is important in the form of the moment of inertia. We diagonalize the moment of inertia for all objects by pre-aligning the initial configuration along their principal axes. Thus, we represent an object's moment of inertia with three degrees of freedom along the diagonal of $\mathbf{I}_0$,

$$\mathbf{I}_0 = \frac{1}{m} diag \begin{pmatrix} i_{0,0} & i_{1,1} & i_{2,2} \end{pmatrix}, \tag{5.3}$$

$$\mathbf{I} = \mathbf{R_q} \mathbf{I}_0 \mathbf{R_q}^{-1}. \tag{5.4}$$

Starting from an input video, we first automatically extract a dense set of image-space centroids with associated depth values and semi-automatically generate a sparse set of orientations. While we formulate the problem in 3D, we will later on minimize the projected error in screen space.

Our goal is to compute: a parametrization of the trajectories of the two bodies a and b, their relative mass, their pre- and post-collision velocities (linear as well as angular), the time of collision, the collision impulse, and the coefficient of restitution. These quantities should match the observed data from the input video as well as possible. We will describe our approach to incorporate these unknowns in our solution in this order. As the image data is potentially unreliable and noisy, we do not rely purely on the corresponding data-terms in our calculations, but use the physical laws as regularizers. One of our central contributions is identifying and incorporating the relevant physical constraints.

5.2.3 Center of Mass Trajectories

A well known fact is that the trajectory of the center of mass of an object experiencing a constant acceleration is given by a parabola. This description becomes invalid at the time of the collision t^c, but it is an excellent model for the trajectories before and after, as long as effects such as aerodynamic drag are negligible. For the two objects, with pre- and post-collision trajectories, we thus parametrize and extract four 3D parabolas from

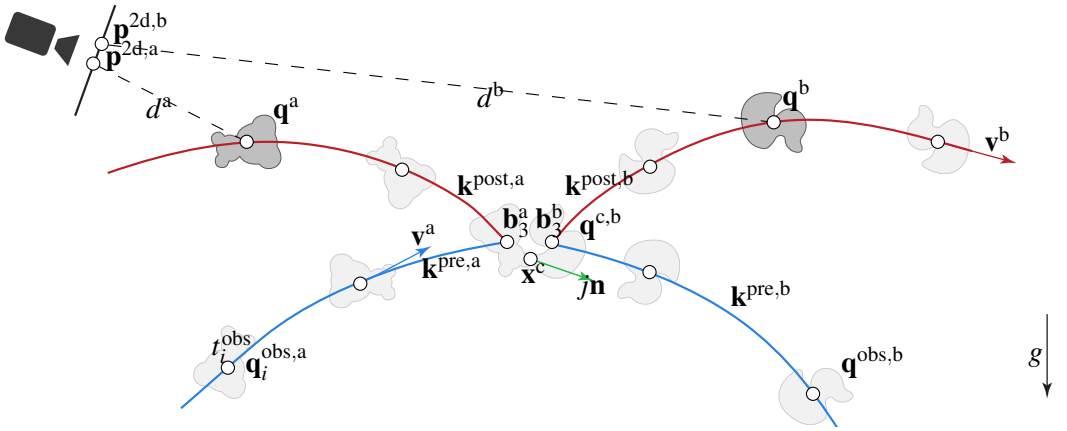


Figure 5.3: Modeling two-body collisions: We extract dense position estimates from a video input and augment it with sparse semi-automatically generated orientations. These observations are linked with laws of physics: the individual trajectories should be parabolas as we assume the bodies to have ballistic motion, and they are coupled at the (unknown) collision point based on conservation laws.

the inputs. We parametrize each parabola in 3D space with

$$\mathbf{p}(t) = \mathbf{R} \begin{pmatrix} b_1(t-t^c) \\ -\frac{g}{2}(t-t^c)^2 + b_2(t-t^c) \\ 0 \end{pmatrix} + \mathbf{b}_3, \tag{5.5}$$

where b_1, b_2 parametrize the curve over time, the vector $\mathbf{b}_3$ determines its offset, and the rotation matrix $\mathbf{R}$ its orientation. We shift the curve in time by the collision time t^c, so that $\mathbf{p}(t^c) = \mathbf{b}_3$.

Moreover, the different parabolas can only rotate about the axis of gravity, which, without loss of generality, we assume to be the y-axis in our formulation. We encode this premise as a global rotation with two degrees of freedom shared by all four parabolas and individual rotations around the gravity direction. As a consequence, we construct a rotation matrix $\mathbf{R}$ as a sequence of Euler angle rotations. We chose the proper Euler angle representation Y,X,Y, the angles of which we denote with β_{y0}, β_x, and β_{y1}, respectively. We denote the resulting rotation matrix by $\mathbf{R}_{\beta_{y0},\beta x,\beta_{y1}}$. Note that β_{y0} is different for every parabola, while the other two are shared, rotating the common gravity vector.

Further, the two parabolas for an object coincide in one point at the time of collision, allowing us to express both parabolas uniquely with a single $\mathbf{b}_3$ offset. In total, that leaves us with two global angles, four individual angles, two offsets, and four times b_1, b_2 as unknowns. The geometric setup of the four parabolas is illustrated in Figure 5.3.

We directly estimate the linear velocity from these curves with the temporal derivative of Equation (5.5). The most important velocity in our setting is the velocity at collision time t^c:

$$\mathbf{v}(t^c) = \mathbf{R}\begin{pmatrix} b_1 & b_2 - gt^c & 0 \end{pmatrix}^T \tag{5.6}$$

We now describe the angular aspects of our motion modeling.

5.2.4 Orientation and Angular Momentum

While the angular velocity of a rigid body can change without external forces, its angular momentum remains constant (see [83]). Thus, for each of the four trajectories, we calculate an angular momentum $\mathbf{k}$ that best explains the orientations observed over time. Angular momentum and velocity relate to each other with:

$$\omega(t) = \mathbf{I}^{-1}\mathbf{k}(t) = \mathbf{R}_\mathbf{q}(t)\mathbf{I}_0^{-1}\mathbf{R}_\mathbf{q}^{-1}(t)\mathbf{k}(t) \tag{5.7}$$

Once we know the angular momentum at time t, we can compute the corresponding angular velocity ω, and use it to integrate an orientation forward in time. With ω as the imaginary part of a quaternion, an Euler step is given by:

$$\mathbf{q}(t+\Delta t) = \mathbf{q}(t) + \frac{\Delta t}{2}\begin{pmatrix} 0 \\ \omega \end{pmatrix} \otimes \mathbf{q}(t) \tag{5.8}$$

where $\otimes$ denotes the quaternion product, and we normalize $\mathbf{q}(t+\Delta t)$ after each step.

Similar to the shared offset of the parabolas, we introduce an orientation $\mathbf{q}^c$ for each body at collision time that is shared by the two parabola segments. Our goal is not to use one of the input orientations directly, due to their potential unreliability. Instead we find an improved orientation that, together with an angular momentum $\mathbf{k}$, explains the observed orientations as well as possible. Starting with $\mathbf{q}^c$ at collision time t^c we integrate this orientation backward or forward to time t with explicit integration. We perform a series of integration steps with Equation (5.8) from t^c to a given time t. This yields the orientation of body a or b at any instance in time based on the estimated orientation at collision time. We will later on use this step to calculate the estimated orientation at different times, based on our current estimate for the collision orientation

and object motion. Thus it will later on serve to guide the optimization towards the input orientations.

5.2.5 Energy Limiting

When gravity is the only external force present, the sum of kinetic and potential energy (for height h) remains constant as:

$$E = \tfrac{1}{2} m\, \mathbf{v}^T \mathbf{v} + \tfrac{1}{2} \boldsymbol{\omega}^T \mathbf{I} \boldsymbol{\omega} + mgh. \tag{5.9}$$

Upon collision, a portion of this energy dissipates. This is typically modeled with the aforementioned coefficient of restitution. As it is one of the central parameters that we want to retrieve, we cannot directly impose constraints conserving the energy in our system. However, assuming the collision is an instantaneous event, we can ensure that the post-collision energy is less or equal to the pre-collision energy. As position and orientation do not change at the time of collision, the potential energy remains the same. Thus, our formulation focuses on the kinetic energy, and we will later on ensure that it does not increase after the collision.

5.2.6 Conservation of Momentum

Throughout the objects' trajectories, angular momentum is conserved in the absence of external torques. Linear momentum is only conserved in the direction orthogonal to gravity. Instantaneous rigid body collisions fully preserve both total linear and total angular momentum of the objects involved. As such, the equations for an impulse-based collision are ideal to couple the four separate trajectories at the time of collision. We have already exploited the fact that the two parabolas for each object have to connect in the respective collision positions, additionally, the conservation laws for momentum enable the coupling of the two objects' trajectories.

Conservation of linear momentum means that the sum of linear momenta before and after collision has to be equal. Since we cannot recover the absolute mass, we set the mass of body a to 1.0, and introduce a mass ratio $m^{\mathrm{b,a}} = m^{\mathrm{b}}/m^{\mathrm{a}}$, (we will revisit the case of a potentially infinitely heavy object in Section 5.4.2). Formulating the equations

in terms of $m^{\mathrm{b,a}}$ yields

$$\mathbf{v}^{\mathrm{pre,a}}(t^{\mathrm{c}}) + \mathbf{v}^{\mathrm{pre,b}}(t^{\mathrm{c}})m^{\mathrm{b,a}} = \mathbf{v}^{\mathrm{post,a}}(t^{\mathrm{c}}) + \mathbf{v}^{\mathrm{post,b}}(t^{\mathrm{c}})m^{\mathrm{b,a}} \ . \tag{5.10}$$

As the velocities are determined from the parabolas, this equation ensures that these curves do not create or lose linear momentum, and effectively couple the planes of the four parabolas.

We similarly ensure that the angular momentum we compute does not violate conservation of angular momentum. For the angular momenta, we have to consider their sum, and additionally the instantaneous angular momentum around the origin:

$$\begin{aligned} &\mathbf{p}^{\mathrm{a}} \times \mathbf{v}^{\mathrm{pre,a}} + \mathbf{k}^{\mathrm{pre,a}} + m^{\mathrm{b,a}}\mathbf{p}^{\mathrm{b}} \times \mathbf{v}^{\mathrm{pre,b}} + \mathbf{k}^{\mathrm{pre,b}} = \\ &\mathbf{p}^{\mathrm{a}} \times \mathbf{v}^{\mathrm{post,a}} + \mathbf{k}^{\mathrm{post,a}} + m^{\mathrm{b,a}}\mathbf{p}^{\mathrm{b}} \times \mathbf{v}^{\mathrm{post,b}} + \mathbf{k}^{\mathrm{post,b}} \end{aligned} \tag{5.11}$$

Note, that we use $m^{\mathrm{b,a}}$ to compute $\mathbf{I}_0^{\mathrm{b}}$ and $\mathbf{k}^{\mathrm{b}}$ using Equation (5.3). All quantities in Equation (5.11) are evaluated at collision time t^{c}. The two equations (5.10) and (5.11) constrain our solution space to a physically realistic one, and provide a first coupling between objects a and b. We improve this coupling with the help of the impulse equations from Section 5.2.1. The equations for linear impulse (Equation (5.1)) express the exchange of momentum between the two bodies relative to their mass:

$$\begin{aligned} \mathbf{v}^{\mathrm{post,a}}(t^{\mathrm{c}}) &= \mathbf{v}^{\mathrm{pre,a}}(t^{\mathrm{c}}) + \frac{j\mathbf{n}}{1} \ , \\ \mathbf{v}^{\mathrm{post,b}}(t^{\mathrm{c}}) &= \mathbf{v}^{\mathrm{pre,b}}(t^{\mathrm{c}}) - \frac{j\mathbf{n}}{m^{\mathrm{b,a}}} \ . \end{aligned} \tag{5.12}$$

This equation turns out to be crucial for retrieving the mass ratio $m^{\mathrm{b,a}}$ during the combined optimization, which we will detail below.

The angular impulse equations allow us to enforce a shared collision point $\mathbf{x}^{\mathrm{c}}$. Equation (5.2) connects the angular momenta of the two bodies via the relative offset of the collision point:

$$\begin{aligned} \mathbf{k}^{\mathrm{post,a}} &= \mathbf{k}^{\mathrm{pre,a}} + ((\mathbf{x}^{\mathrm{c}} - \mathbf{b}_3^{\mathrm{a}}) \times j\mathbf{n}) \\ \mathbf{k}^{\mathrm{post,b}} &= \mathbf{k}^{\mathrm{pre,b}} - ((\mathbf{x}^{\mathrm{c}} - \mathbf{b}_3^{\mathrm{b}}) \times j\mathbf{n}) \ . \end{aligned} \tag{5.13}$$

We complete the set of necessary equations by computing the relative velocities at the collision point to compute the coefficient of restitution. We only need to consider the velocity components along the collision normal direction:

$$c = -\frac{\left(\hat{\mathbf{v}}^{\text{post,a}} - \hat{\mathbf{v}}^{\text{post,b}}\right) \cdot \mathbf{n}}{\left(\hat{\mathbf{v}}^{\text{pre,a}} - \hat{\mathbf{v}}^{\text{pre,b}}\right) \cdot \mathbf{n}}, \tag{5.14}$$

with $\hat{\mathbf{v}} = \mathbf{v}(t^{\text{c}}) + \boldsymbol{\omega}(t^{\text{c}}) \times (\mathbf{x}^{\text{c}} - \mathbf{p}(t^{\text{c}}))$ and $\boldsymbol{\omega}(t^{\text{c}}) = \mathbf{I}^{-1}(t^{\text{c}})\,\mathbf{k}(t^{\text{c}})$. With these equations at hand we now formulate an optimization problem to compute the unknown quantities as described next.

5.3 Method

Our method proceeds in three stages: First, we estimate center of mass positions based on the input video. Second, we initialize orientations using a differential raytracer, with user input for ambiguous situations. Third, in the main step, we take these position and orientation estimates to retrieve the physical parameters of the rigid body motion and collision. For the initialization, we use a stripped-down version of our main optimization step. We first explain the main step of the algorithm, before detailing the two initialization stages.

5.3.1 Parameter Estimation

The quantities described in Section 5.2 can be divided into those related to the input data and those related to the underlying physics. In terms of unknowns, we distinguish between the parameters of the four trajectory segments of the two objects, which determine position and velocities, and the unknowns of the collision event. We will use a superscript * to refer to the four trajectories, *i.e.,* $* \in \{\text{pre-a}, \text{pre-b}, \text{post-a}, \text{post-b}\}$ and $^\diamond$ for objects, *i.e.,* $\diamond \in \{\text{a}, \text{b}\}$.

For each trajectory, we have the unknowns of each parabola. They are b_1^*, b_2^*, β_{y0}^*, while $\mathbf{b}_3^\diamond$ exist once for each body. Also, we have the global angles β_x, β_{y1}, and unknowns for angular momentum exist for each trajectory segment, $\mathbf{k}^*$. Additionally, the unknowns for the collision are: orientation for both objects at collision time $\mathbf{q}^{\text{c},\diamond}$, location $\mathbf{x}^{\text{c}}$, impulse $j\mathbf{n}$, and the mass ratio $m^{\text{b,a}}$, as well as the coefficient of restitution c and the time of the collision t^{c}.

Our optimization uses a dense set of 2D object centroids $\{\mathbf{p}_i^{\mathrm{2d}}\}_{i=1}^N$ with depth values $\{d_i\}_{i=1}^N$ at times $\{t_i^{\mathrm{obs}}\}_{i=1}^N$, and a sparse set of orientations for both bodies $\{\mathbf{q}_j^{\mathrm{obs}}\}_{j=1}^M$ at times $\{t_j^{\mathrm{obs}}\}_{j=1}^M$ from the input video. Later, in Section 5.3.2, we describe how to extract this data from the video. We use between 200 and 350 video frames with position information, and two semi-automatically annotated orientations per parabola segment, thus eight orientations in total. Additionally, we require a bounding box estimate for each body: $\mathbf{s}^{\diamond}$. Finally, we have a set of additional equations from the conservation laws, namely Equations (5.10-5.21) to tie the observations together according to the physical model. They represent our physics-based regularization encoded as additional equations.

As the number of equations from input data and physics is significantly larger than the number of unknowns, we have an over-determined problem. As the equations are non-linear, we compute a solution using non-linear least-squares. We summarize the unknowns in the vector θ for this least-squares optimization, and reformulate the equations to take the form

$$\underset{\theta}{\operatorname{argmin}} \sum_{\tau \in \mathtt{Terms}} w^{\tau} f^{\tau}(\theta) \,, \tag{5.15}$$

where $\mathtt{Terms} = \{\mathtt{mom}, \mathtt{imp}, \mathtt{g}, \mathtt{ke}, \mathtt{CoR}, \mathtt{pxy}, \mathtt{pz}, \mathtt{ori}\}$ and w^{τ} is weighting the corresponding energy term f^{τ}. We now describe these individual energy terms.

Respecting physics. We directly turn the physics constraints in Equations (5.10-5.21) into energy terms by defining the least squares energy to be their residual. We combine the equations for momentum conservation (5.10,5.11) into $f^{\mathtt{mom}}$:

$$\begin{aligned} f^{\mathtt{mom}} = &\frac{1}{2} \| \mathbf{v}^{\mathrm{pre,a}} + \mathbf{v}^{\mathrm{pre,b}} m^{\mathrm{b,a}} - \mathbf{v}^{\mathrm{post,a}} - \mathbf{v}^{\mathrm{post,b}} m^{\mathrm{b,a}} \|^2 \\ &+ \frac{1}{2} \| \mathbf{p}^{\mathrm{a}} \times \mathbf{v}^{\mathrm{pre,a}} + \mathbf{k}^{\mathrm{pre,a}} + m^{\mathrm{b,a}} \mathbf{p}^{\mathrm{b}} \times \mathbf{v}^{\mathrm{pre,b}} + \mathbf{k}^{\mathrm{pre,b}} - \\ &\quad \mathbf{p}^{\mathrm{a}} \times \mathbf{v}^{\mathrm{post,a}} - \mathbf{k}^{\mathrm{post,a}} - m^{\mathrm{b,a}} \mathbf{p}^{\mathrm{b}} \times \mathbf{v}^{\mathrm{post,b}} - \mathbf{k}^{\mathrm{post,b}} \|^2 \,, \end{aligned} \tag{5.16}$$

and the impulse equations (5.12,5.13) into $f^{\texttt{imp}}$:

$$\begin{aligned} f^{\texttt{imp}} = \frac{1}{4}\Big(& \|\mathbf{v}^{\text{pre,a}} + \frac{j\mathbf{n}}{1} - \mathbf{v}^{\text{post,a}}\|^2 + \|\mathbf{v}^{\text{pre,b}} - \frac{j\mathbf{n}}{m^{\text{b,a}}} - \mathbf{v}^{\text{post,b}}\|^2 \\ + & \ \|\mathbf{k}^{\text{pre,a}} + ((\mathbf{x}^{\text{c}} - \mathbf{b}_3^a) \times j\mathbf{n}) - \mathbf{k}^{\text{post,a}}\|^2 \\ + & \ \|\mathbf{k}^{\text{pre,b}} - ((\mathbf{x}^{\text{c}} - \mathbf{b}_3^b) \times j\mathbf{n}) - \mathbf{k}^{\text{post,b}}\|^2 \Big) \ . \end{aligned} \tag{5.17}$$

Both terms are evaluated at collision time t^{c}, omitted for brevity. As our inputs are typically hand-held videos, we do not fully constrain gravity to act along the y-axis, but allow the optimization to slightly deviate, if necessary. For this we define the energy term

$$f^{\texttt{g}} = \left(g - \begin{pmatrix} 0 & 1 & 0 \end{pmatrix} \mathbf{R}_{0,\beta_x,\beta_{y1}} \begin{pmatrix} 0 & g & 0 \end{pmatrix}^T \right)^2 \ . \tag{5.18}$$

Note that β_{y0} is zero here, as this value represents rotation around the axis of gravity. We formulate the kinetic energy limiting term from Equation (5.9) as:

$$f^{\texttt{ke}} \ = \begin{cases} (E_k^{\text{post}} - E_k^{\text{pre}})^2 & \text{for } E_k^{\text{post}} > E_k^{\text{pre}} \\ 0 & \text{otherwise,} \end{cases} \tag{5.19}$$

where pre- and post-collision kinetic energies are evaluated in terms of velocities at the time of collision:

$$\begin{aligned} \forall \times \in \{\text{pre}, \text{post}\} \quad E_k^{\times} = & \quad \mathbf{v}^{\times,\text{a}}(t^{\text{c}})^2 \ + \ \boldsymbol{\omega}^{\times,\text{a}}(t^{\text{c}})^T \mathbf{I}^{\text{a}}(t^{\text{c}}) \ \boldsymbol{\omega}^{\times,\text{a}}(t^{\text{c}}) \\ & + \ m^{\text{b,a}} \mathbf{v}^{\times,\text{b}}(t^{\text{c}})^2 \ + \ \boldsymbol{\omega}^{\times,\text{b}}(t^{\text{c}})^T \mathbf{I}^{\text{b}}(t^{\text{c}}) \ \boldsymbol{\omega}^{\times,\text{b}}(t^{\text{c}}) \ . \end{aligned} \tag{5.20}$$

Finally, to disallow energy creation, we directly constrain the c estimate to physical range of zero to one. For this we include a term

$$f^{\texttt{CoR}} = \begin{cases} c^2 & \text{for } c < 0 \\ (c-1)^2 & \text{for } c > 1 \\ 0 & \text{otherwise ,} \end{cases} \tag{5.21}$$

with c calculated using Equation (5.21).

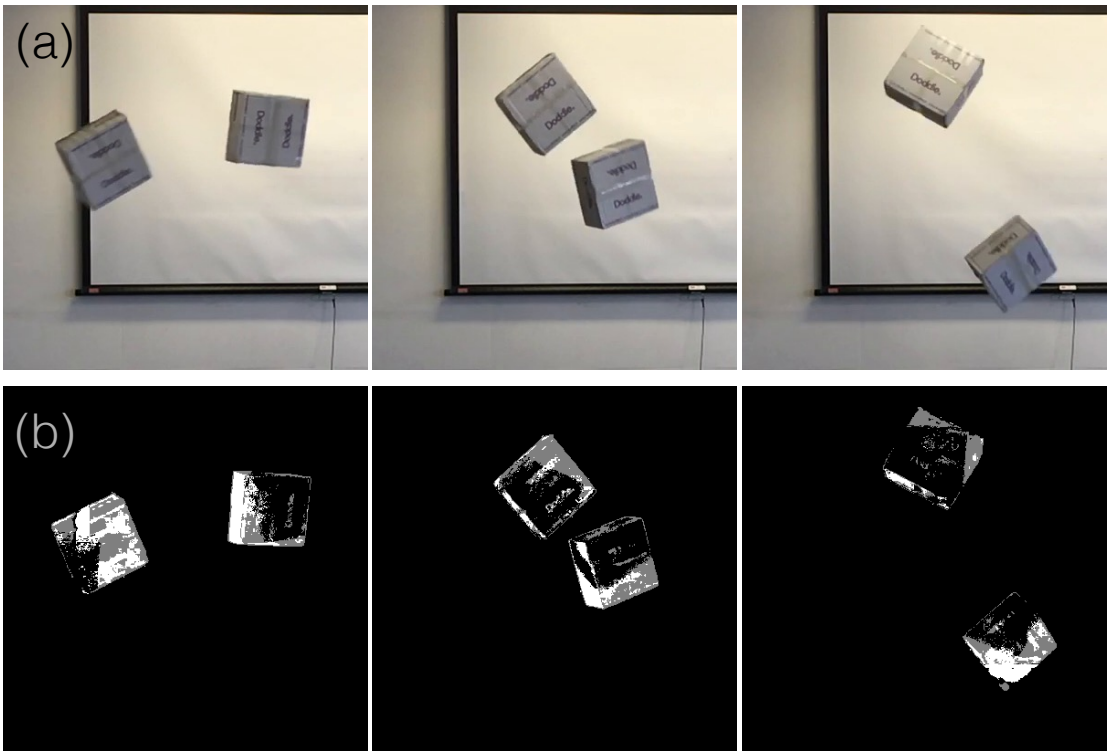


Figure 5.4: In order to automatically retrieve measurements of the object centroids, we first post-process the input video (a) to subtract the background. This step yields approximate, and noisy image regions for the objects (b).

Aligning to data. Next, we formulate the data terms. Based on Equation (5.5) we add terms for position fidelity. The notation is slightly complicated by the fact that we do not directly compute the difference of two 3D positions, but instead separately consider screen space and depth, each of which we will detail below.

For the screen space positions $\mathbf{p}_i^{2\mathrm{d},*}$ (each at time t_i^{obs}) that were extracted from the video, we add an energy term penalizing screen-space deviations of the corresponding parabola with

$$\forall * \in \{\text{pre-a}, \text{pre-b}, \text{post-a}, \text{post-b}\} \;\; \mathbf{p}_i^*(t_i^{\mathrm{obs}}) = \begin{pmatrix} b_1^*(t_i^{\mathrm{obs}} - t^{\mathrm{c}}) \\ -\frac{g}{2}(t_i^{\mathrm{obs}} - t^{\mathrm{c}})^2 + b_2^*(t_i^{\mathrm{obs}} - t^{\mathrm{c}}) \\ 0 \end{pmatrix} + \mathbf{b}_3^{\diamond}$$

$$s.\ t.\ \diamond = \begin{cases} \mathrm{a} & \mathit{if}\ * \in \{\text{pre-a}, \text{post-a}\} \\ \mathrm{b} & \mathit{if}\ * \in \{\text{pre-b}, \text{post-b}\} \end{cases}, \quad \text{and} \quad f^{\mathtt{pxy}} = \frac{1}{N}\sum_{i=1}^{N} \|\mathbf{P}\,\mathbf{p}_i^*(t_i^{\mathrm{obs}}) - \mathbf{p}_i^{2\mathrm{d},*}\|^2 \,, \tag{5.22}$$

where $\mathbf{P}$ is the camera projection matrix and $N = |\{\mathbf{p}_i^{2d,*}\}_{*\in\{\text{pre-a,pre-b,post-a,post-b}\}}|$. Note that $\mathbf{b}_3^{\text{pre},\diamond} \equiv \mathbf{b}_3^{\text{post},\diamond}$ is shared among pre- and post-collision parabolas of the same object.

For the depth, we compute the energy terms with

$$f^{\texttt{pz}} = \frac{1}{N}\sum_{i=1}^{N} \| \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} \mathbf{p}_i^*(t_i^{\text{obs}}) - d_i \|_\varepsilon \ , \tag{5.23}$$

where $\|\cdot\|_\varepsilon$ denotes a robust *Huber* [68] loss function to allow for well-supported changes in depth. Specifically,

$$\|x\|_\varepsilon = \begin{cases} x^2/(2\varepsilon) & \text{if } x^2 \leq \varepsilon \\ |x| - \varepsilon/2 & \text{otherwise.} \end{cases} \tag{5.24}$$

The energy terms for the orientations are straight-forward to add based on Equation (5.8):

$$f_j^{\texttt{ori}} = \frac{1}{M}\sum_{j=1}^{M} \|\mathbf{q}^*(t_j^{\text{obs}}) - \mathbf{q}_j^{\text{obs},*}\|^2 \ , \tag{5.25}$$

where $M = |\{\mathbf{q}_j^{\text{obs},*}\}_{*\in\{\text{pre-a,pre-b,post-a,post-b}\}}|$, and $\mathbf{q}^*(t_j^{\text{obs}})$ is expressed as a series of explicit integration steps starting from $\mathbf{q}^{c,\diamond}, \diamond \in \{\text{a, b}\}$. We use the analytic equations for cuboids to calculate the moment of inertia for our bodies. We pre-align their principal moments with the coordinate axes using SVD on the scanned models during a pre-computation step. We then scale it with the masses of the objects, *i.e.,* 1.0 and $m^{\text{b,a}}$. For a specific instance in time, $\mathbf{I}_0$ is transformed with the orientation at this time according to Equation (5.4).

Collision point. As our optimization has to navigate a highly non-linear energy landscape, a good initialization is crucial (see Sections 5.3.2 and 5.3.3). Additionally, we found that our optimization profits from constraining the collision point at first. We thus run our full optimization twice: first (Algorithm 3, line 25) with a collision point $\mathbf{x}^{\text{c}} = (\mathbf{b}_3^{\text{a}} + \mathbf{b}_3^{\text{b}})/2$, *i.e.,* constrained to lie at the midpoint of the collision positions of both objects. Removing these degrees of freedom at first increases robustness, especially for cases with lower quality data.

For the second run, we release the collision point to compute the final parameters of the collision (Algorithm 3, line 26). While mass ratio and c have been computed

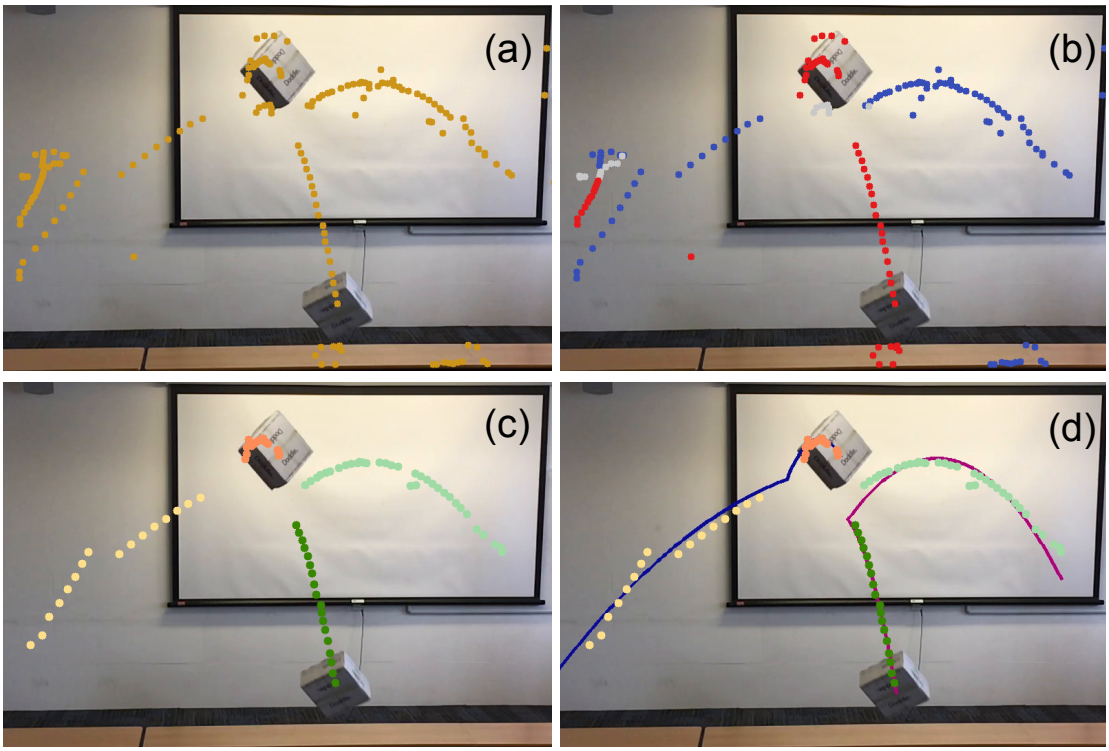


Figure 5.5: The steps of our position initialization: (a) object region centroids; (b) initial labeling: pre-collision in blue, post-collision in red, and discarded points in gray; (c) re-assignment, and (d) final parabolas: object A in blue, object B in purple.

during the solve, we evaluate the final positions and orientations from our continuous, analytical parametrization.

In the following, we explain how to obtain measurements of screen space object centroids and 3D object orientations from the video.

5.3.2 Initializing Positions

We simultaneously estimate screen space object centroid measurements, assign them to trajectory segments, and initialize the collision in 3D. We assume that both objects are roughly equidistant to the camera at all times and that their 3D bounding box estimates from the initial scans are available. Essentially, we use the known gravitational acceleration and video frame rate to position the collision in 3D.

Outline. First, we calculate image region candidates for the reprojection of the objects in each frame. Then, we perform a RANSAC labeling involving several, modified parabola fits to compute temporally consistent assignments of the image region centroids to the four trajectory segments (see line 9 in Algorithm 3) and to identify

Algorithm 3: SMASH

INITIALIZEPOSITIONS () /* Sec. 5.3.2 */
INITIALIZEORIENTATIONS () /* Sec. 5.3.3 */
RECONSTRUCTCOLLISION () /* Sec. 5.3.1 */

Function INITIALIZEPOSITIONS /* Sec. 5.3.2 */
Input : video, $\mathbf{s}^{\diamond}$, $\mathbf{P}$, f_{fps}, g
Assumption: $\beta^{*}_{y0} = 0$, $\mathbf{b}^{\text{a}}_{3,z} \equiv \mathbf{b}^{\text{b}}_{3,z}$
$\widetilde{t^{\text{c}}} \leftarrow N/2$
do
  $t^{\text{c}}_{init} \leftarrow \widetilde{t^{\text{c}}}$
  for *e=1* **to** *25* **do** /* RANSAC loop */
    $\{\mathbf{p}^{\text{2d},*}_k\}_{k=1}^{K=0.3N} \leftarrow$ ASSIGNRANDOM(SAMPLE($\{\mathbf{p}^{2d}_i\}$,0.3), t^{c}_{init})
    $\theta \leftarrow$ SOLVE($\{pxy, siz, g\}$, $\{\}$, $\{\mathbf{p}^{2d,*}_k\}_{k=1}^{K}$)
    $\{\mathbf{p}^{\text{2d},*}_i\}_{i=1}^{N} \leftarrow$ ASSIGNGREEDY($\{\mathbf{p}^{2d}_i\}$, θ)
    $\theta \leftarrow$ SOLVE($\{pxy, siz, g\}$, $\{\}$, $\{\mathbf{p}^{2d,*}_i\}_{i=1}^{N}$)
    $\hat{\theta} \leftarrow$ COMPARE($\hat{\theta}$, SCORE(θ, $\{\mathbf{p}^{2d,*}_i\}_{i=1}^{N}$))
while $\widetilde{t^{c}} \leq \lfloor t^{c}_{init} \rfloor$ **or** $\widetilde{t^{c}} \geq \lceil t^{c}_{init} \rceil$
Output : $\{\mathbf{p}^{\text{2d},*}_i\}_{i=1}^{N}, \{d^{*}_i\}_{i=1}^{N}, \{t^{\text{obs}}_i\}_{i=1}^{N}, t^{\text{c}}, \widetilde{b^{*}_1}, \widetilde{b^{*}_2}, \widetilde{\mathbf{b}^{\diamond}_3}, \widetilde{\beta^{*}_{y0}}, \widetilde{\beta_x}, \widetilde{\beta_{y1}}$

Function INITIALIZEORIENTATIONS /* Sec. 5.3.3 */
Input : video, $\{\mathbf{p}^{\text{2d},*}_j\}_{j=1}^{M}$, $\{d^{*}_j\}_{j=1}^{M}$, $\{t^{\text{obs}}_j\}_{j=1}^{M}$, scan$^{\diamond}$
Output : $\{\mathbf{q}^{\text{obs},*}_j\}_{j=1}^{M} \leftarrow$ OPENDR(*Input*)

Function RECONSTRUCTCOLLISION /* Sec. 5.3.1 */
Input : $\{\mathbf{p}^{\text{2d},*}_i\}$,$\{z^{\text{2d},*}_i\}$, $\{\mathbf{q}^{\text{obs},*}_j\}$, $\{t^{\text{obs}}_i\}$, $\mathbf{s}^{\diamond}$, $\mathbf{P}$, f_{fps}, g, t^{c}, $\widetilde{b^{*}_1}$, $\widetilde{b^{*}_2}$, $\widetilde{\mathbf{b}^{\diamond}_3}$, $\widetilde{\beta^{*}_{y0}}$, $\widetilde{\beta_x}$, $\widetilde{\beta_{y1}}$
for $\diamond \in \{a, b\}$ **do**
  $\{j^{\text{pre},\diamond}, j^{\text{post},\diamond}\} \leftarrow j \mid t^{\text{obs},\diamond}_j$ closest before/after t^{c}
  $\mathbf{q}^{\text{c},\diamond} \leftarrow$ SLERP($\mathbf{q}^{obs,\diamond}_{j^{pre,\diamond}}, \mathbf{q}^{obs,\diamond}_{j^{post,\diamond}}, t^{obs,\diamond}_{j^{pre,\diamond}}, t^{c}$)
  for $j^{*} \in \{j^{pre,\diamond}, j^{post,\diamond}\}$ **do**
    $\begin{pmatrix} \cdot & \widetilde{\omega}^{*} \end{pmatrix}^{T} \leftarrow \left(2(\mathbf{q}^{\text{obs},*}_{j^{*}} - \mathbf{q}^{\text{c},\diamond})/(t^{\text{obs}}_{j^{*}} - t^{\text{c}})\right) \otimes (\mathbf{q}^{\text{c},\diamond})^{-1}$
    $\mathbf{k}^{*} \leftarrow \mathbf{R}_{\mathbf{q}^{\text{c},\diamond}} \mathbf{I}_0 \mathbf{R}^{-1}_{\mathbf{q}^{\text{c},\diamond}} \widetilde{\omega}^{*}$
$\mathbf{x}^{\text{c}}_{init} \leftarrow (\widetilde{\mathbf{b}^{\text{a}}_3} + \widetilde{\mathbf{b}^{\text{b}}_3})/2$
$\theta \leftarrow$ SOLVE($\{mom, imp, g, ke, CoR, pxy, pz, ori\}$, $\{\mathbf{x}^{c}\}$)
$\theta \leftarrow$ SOLVE($\{mom, imp, g, ke, CoR, pxy, pz, ori\}$, $\{\}$)
Output : $b^{*}_1, b^{*}_2, \mathbf{b}^{\diamond}_3, \beta^{*}_{y0}, \beta_x, \beta_{y1}, \mathbf{q}^{\text{c},\diamond}, \mathbf{k}^{*}$

Function SOLVE (Terms, FixedVars, Data = *All*)
return $\theta =$ solve Eq. (5.15) | Terms, FixedVars, Data

false-positive image regions. Finally, we iterate this process in a loop to determine the approximate time of collision.

Image region calculation. We estimate candidate center of mass positions in image space. We use a standard background subtraction pipeline consisting of Gaussian mixture modeling [196] and connected contour detection [157]. Figure 5.4 shows the output of the background subtraction step for three frames of a collision.

RANSAC labeling. We choose the middle of the video sequence as initial guess for the collision time t^{c}, more specifically, the midpoint in time between the two consecutive images. We then assign a randomly chosen centroid position from each video frame before and after t^{c} to the pre- and post-collision parabolas respectively (line 10 in Algorithm 3). We omit $f_{\mathrm{fps}}/10$ frames (24 at 240fps) around collision time, as we found these to be particularly unreliable due to the merged silhouettes.

We perform a modified non-linear least squares fit to calculate the parabola parameters from Section 5.2.3 using the assigned screen space centroids, only considering the 2D position fidelity term (Equation (5.22)) and a new size penalty term:

$$f^{\mathtt{siz}} = \frac{1}{N}\sum_{i=1}^{N}\left(\|\mathbf{P}(\mathbf{p}^*(t_i)+\frac{\mathbf{d}}{2}) - \mathbf{P}(\mathbf{p}^*(t_i)-\frac{\mathbf{d}}{2})\| - s_i^{2\mathrm{d}}\right)^2, \tag{5.26}$$

where $\mathbf{p}^*(t)$ is the position on a parabola at time t, $\mathbf{d}$ is the 3D diagonal of the scanned object, $s^{2\mathrm{d}}$ denotes the size of a circle fit to the image region corresponding to the selected centroid for this frame. This term pushes a parabola position towards a depth value so that the projected object diagonals match the image region sizes.

For this optimization, we share the z-coordinate of $\mathbf{b}_3$ between the two objects to ensure the objects meet in space. We also fix the parabola rotations around y to 0 ($\beta^*_{y0} = 0$) to stabilize the non-linear solve by constraining the space of solutions to be parallel to the camera plane. The subsequent main optimization step will refine these estimates.

The parabola fit allows us to greedily assign centroids to trajectory segments or label them as outliers (line 12 in Algorithm 3).

To evaluate the quality of the obtained fit, we compute a score based on the residual of the parabola fit optimization, in conjunction with a SIFT feature distance f^{Im} to measure the similarity of the selected image regions. For the former, we use the screen-space, area-weighted centroid-to-parabola distances d_{para} from Equation (5.22), multiplied with f^{inlier}, the ratio of centroids correctly explained (with an inlier distance

equal to the median size of the identified regions). The image content score is calculated with a SIFT descriptor distance of the object regions between consecutive frames. The final score is weighted sum of these terms: $d_{\text{para}} f^{\text{inlier}} + w^{\text{Im}} f^{\text{Im}}$, see line 14 in Algorithm 3. The fit with the lowest score is used to store the final set of screen-space centroids $\{\mathbf{p}_i^{2\text{d},*}\}_{i=1}^N$, which are passed on to the main optimization stage. In addition to the 2D positions, we store the depth $\{d_i^*\}_{i=1}^N$ along each of the view rays for our depth terms (Equation (5.23)), and their assignments to the trajectory segments. Figure 5.5 visually illustrates the steps of a single RANSAC iteration, typically converging in about 25 iterations.

Collision time refinement. Note that the optimization above computes an improved value for t^{c} based on the intersection of the parabolas in Equation (5.22). Initially, t^{c} is set to the midpoint of a pair of video frames. If t^{c} shifts before the first image of the pair, or to a point in time after the second one, we restart the calculation after selecting the previous, or next pair of frames, respectively. Their midpoint is the new initial guess for t^{c}, and we re-run the outlier detection, as described above. As this optimization runs very fast, we found that a sequential search for the correct collision time along the video works very well. Alternatively, a bisection search could be easily implemented.

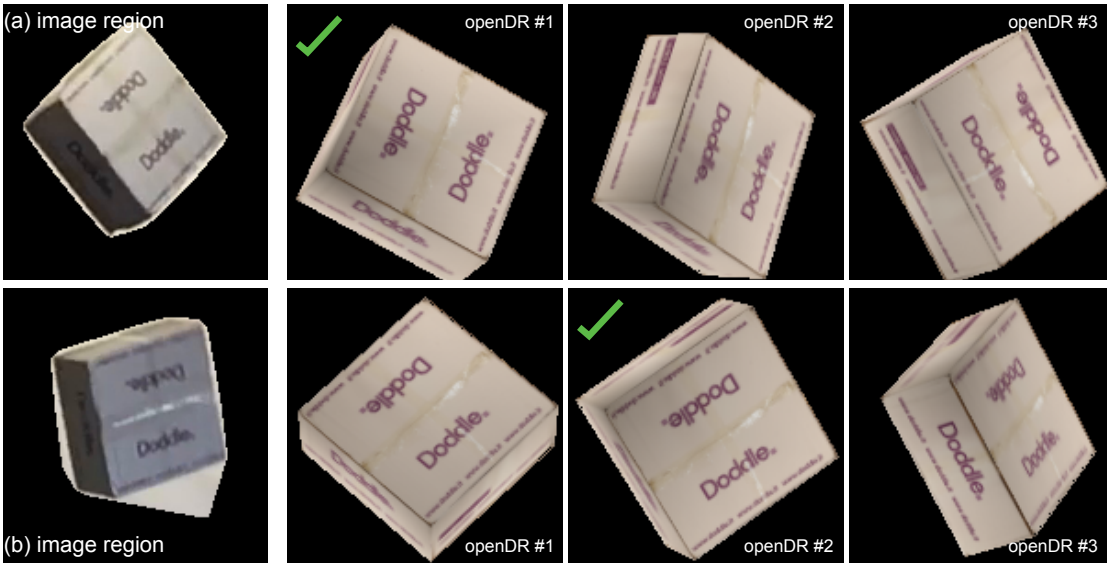


Figure 5.6: Two examples of the image content (left) and the resulting candidate orientations (right). Green check marks highlight the selected orientations.

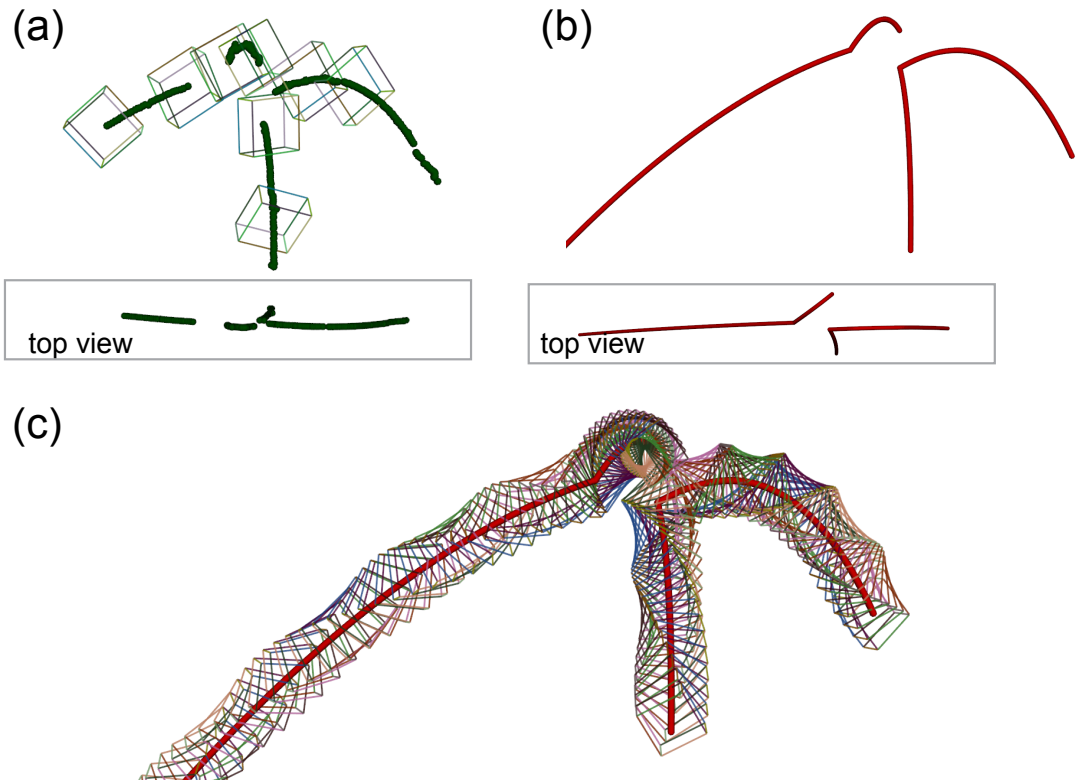


Figure 5.7: a) the result of our initialization: labeled centroids and approximate depth values (green dots) together with the initialized orientations (green cuboids); b) 3D parabolas resulting from our full optimization (red curves), and c) motion over time (colored cuboids).

5.3.3 Initializing Orientations

The previous initialization step yields a first set of 3D positions in time. We can render the scanned objects over the video on these trajectories to initialize the orientations, assuming that 3D scans of the filmed objects are available for this step. Using a differential renderer [96] and the object scans, we perform a first alignment of each object starting from 512 viewpoints uniformly sampled on a sphere. However, due to illumination variations we found the video information to be typically too ambiguous to automatically retrieve the correct orientations for all cases. Hence, we let a user check, and if necessary correct these alignments for two points in time for each parabola segment. This step takes less than a minute per collision sequence. An example of the generated candidate orientations can be seen in Figure 5.6.

The labelled screen-space centroid positions $\{\mathbf{p}_i^{2\mathrm{d},*}\}_{i=1}^{N}$ with depths $\{d_i^*\}_{i=1}^{N}$, and the $M = 8$ input orientations $\{\mathbf{q}_j^{\mathrm{obs},*}\}_{j=1}^{M}$ are the input data for our full parameter estimation step (Section 5.3.1).

Scene (#frames)	BgFg	IniPos	IniOri	Solve 1	Solve 2
box-1 (217)	5.68s	2m 6s	18m 46s	0.55s	0.08s
box-2 (191)	4.51s	1m 51s	17m 47s	0.23s	0.03s
box-3 (236)	8.12s	2m 0s	19m 2s	0.36s	0.05s
helmet-duck (316)	8.63s	1m 40s	24m 41s	0.98s	0.12s
duck-eleph. (204)	5.53s	1m 57s	25m 55s	3.08s	0.18s
duck-rugby (316)	11.86s	1m 43s	25m 11s	0.65s	0.36s

Table 5.2: Timings on a system with Intel i7-4700MQ and Nvidia Geforce GTX 770M. The second and third columns describe Section 5.3.2, the fourth column Section 5.3.3. The rightmost two columns reference lines 25 and 26 of Algorithm 3.

5.3.4 Implementation

We use CERES [1] for solving our non-linear least squares formulation, with sparse-Cholesky as linear solver type, and energy derivatives from automatic differentiation. Figure 5.7 shows an example result of the initialization, and the objects' recovered 3D positions and orientations.

Due to the non-linearity of the angular motion in Equation (5.8) we typically use four integration steps per frame. Note that the sequence of integration steps is taken into account during the automatic differentiation performed in the solver. While higher-order integrators or even closed-form Poinsot solutions would be available to express a rigid body orientation over time, we found that explicit integration yields stable solutions in practice, and leave alternatives for future work. We use f_{fps} to convert time t to be measured in frames.

In our formulation above, we took care to minimize the number of free variables and energy terms. It turned out to be crucial for a good solution to use a formulation with shared variables, instead of enforcing equalities with highly weighted penalty terms. For example, instead of adding a term to minimize $\mathbf{b}_3^{\text{pre,a}} - \mathbf{b}_3^{\text{post,a}}$, we formulate the parabolas with a shared variable, which inherently enforces the equality of the offsets. Likewise, equivalent angles and orientations are implemented with shared variables instead of additional penalty terms. See Table 5.2 for runtimes. We have made code and data available online[1].

[1] `http://geometry.cs.ucl.ac.uk/projects/2016/smash/`

5.4 Evaluation

We tested the robustness of our method and quantitatively evaluated the accuracy of the estimates. First, we report performance on synthetic data to test the system under different perturbations (Section 5.4.1) and then evaluate the system on a range of real-world sequences of increasing complexity (Section 5.4.2).

Accuracy measures. We evaluated the results as: (i) in a synthetic setup, we compare c and $m^{\mathrm{b,a}}$ estimates against ground truth values used to generate the test sequences; (ii) in real-world setups, we either compare with estimates obtained via other means (*e.g.,* using potential energy), or by evaluating consistency of the estimates across multiple collision sequences involving same object pairs. We use $\tilde{x}$ to denote our estimated value for quantity x, thus an output of the optimization. *E*.g., $\tilde{c}$ for our estimate of a coefficient of restitution c. Where applicable, ground truth values *measured* by other means will be denoted with an m subscript, *e.g.,* c_m.

Weights. In *all* the following tests we used a fixed set of weights: $w^{\mathtt{mom}} = w^{\mathtt{imp}} = 10$, $w^{\mathtt{g}} = w^{\mathtt{ke}} = w^{\mathtt{CoR}} = w^{\mathtt{ori}} = w^{\mathtt{pxy}} = 1$, $w^{\mathtt{pz}} = 0.1$, $\varepsilon = 1$, and for the initialization phase $w^{\mathtt{siz}} = 0.1,\ w^{\mathrm{Im}} = 1/500$.

5.4.1 Validation on Synthetic Data

We tested our system using synthetic data obtained from forward simulations using the BULLET physics engine. We created a scene with two colliding cuboids under gravity and rendered the simulation as input video for our algorithm.

(i) Robustness to initialization errors. We separately tested the effect of errors in 2D centroid and orientation estimation in the initialization stage. We rendered the synthetic sequence at 60fps, initialized position for each time frame (Section 5.3.2), and added increasing amount of uniform noise to the retrieved 2D object centroids before fitting the parabolas. The algorithm provided robust estimates for both c and $m^{\mathrm{b,a}}$ up to perturbation margins of 30% (Figure 5.8(a)) with the noise margin being compared to the average screen-space size of the larger object. We ran 63 experiments with different noise margins ($0-30\%$) and found that our parabola fitting to be robust even under moderate errors on centroid estimates. For perturbations involving orientation annotations (Section 5.3.3), we added noise to each coordinate, and re-normalized the

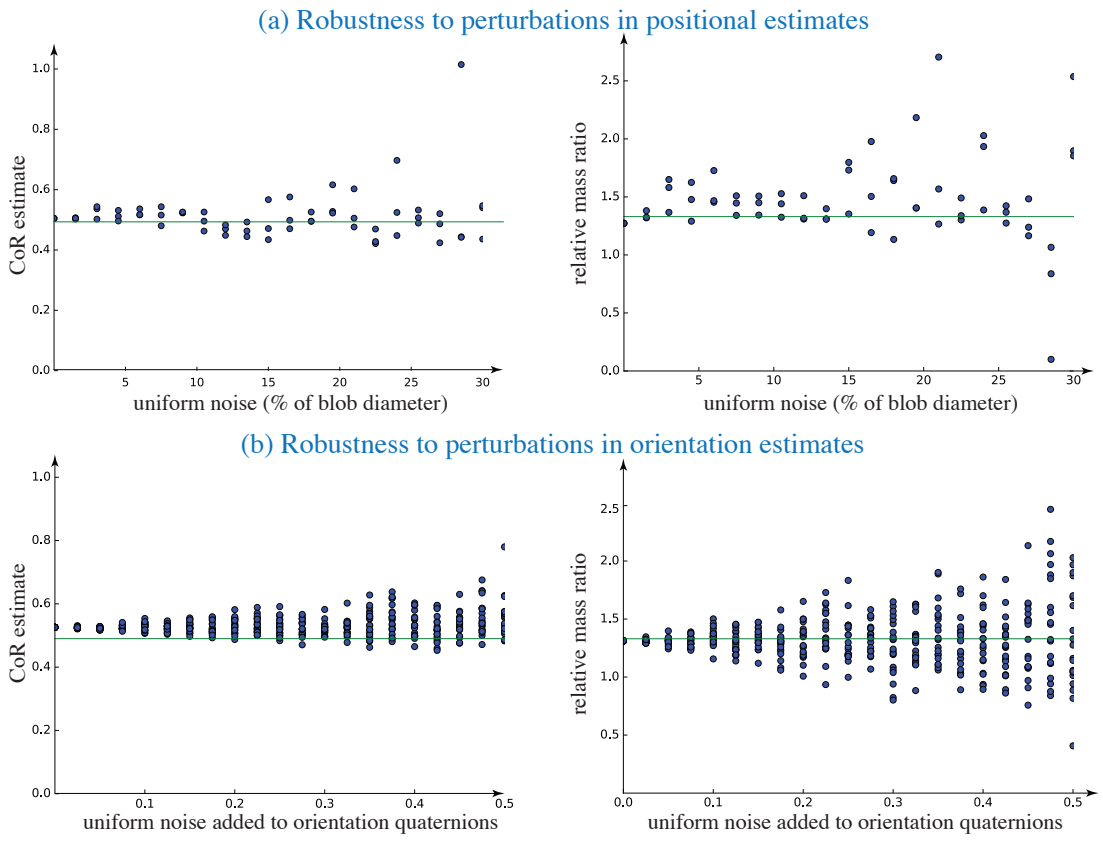


Figure 5.8: Synthetic evaluation of robustness with increasing noise in the position and orientation estimates. Green lines denote ground truth, $c = 0.49$ and $m^{\mathrm{b,a}} = 1.33$.

resultant quaternions. We observed robustness in both $\tilde{c}$ and $\tilde{m}^{\mathrm{b,a}}$ up to perturbation of around 40° around actual orientation estimates (Figure 5.8(b)).

(ii) Effect of physics prior. Next, in an important test, we assessed the contribution of the different physics-based priors in regularizing the optimization. We experimented by activating a selection of the optimization terms and evaluating estimation accuracy (see Figure 5.9). The impulse term has the single most important effect: while just based on the input video, the estimates for c and $m^{\mathrm{b,a}}$ deviate substantially, the impulse-based physics formulation provides valuable regularization resulting in reliable estimates even in the noisy setting. The secondary constraints in the form of bounds on c and kinetic energy further improve the estimation accuracy.

(iii) Effect of annotation frequency. We measured the effect of fewer number of orientation annotations across time on estimated $m^{\mathrm{b,a}}$ and c. Figure 5.11 shows the estimates for increasing annotation intervals. Note that even for the extreme case with an interval of around 20, *i.e.,* with only 4 annotations for a 90 frame sequence our

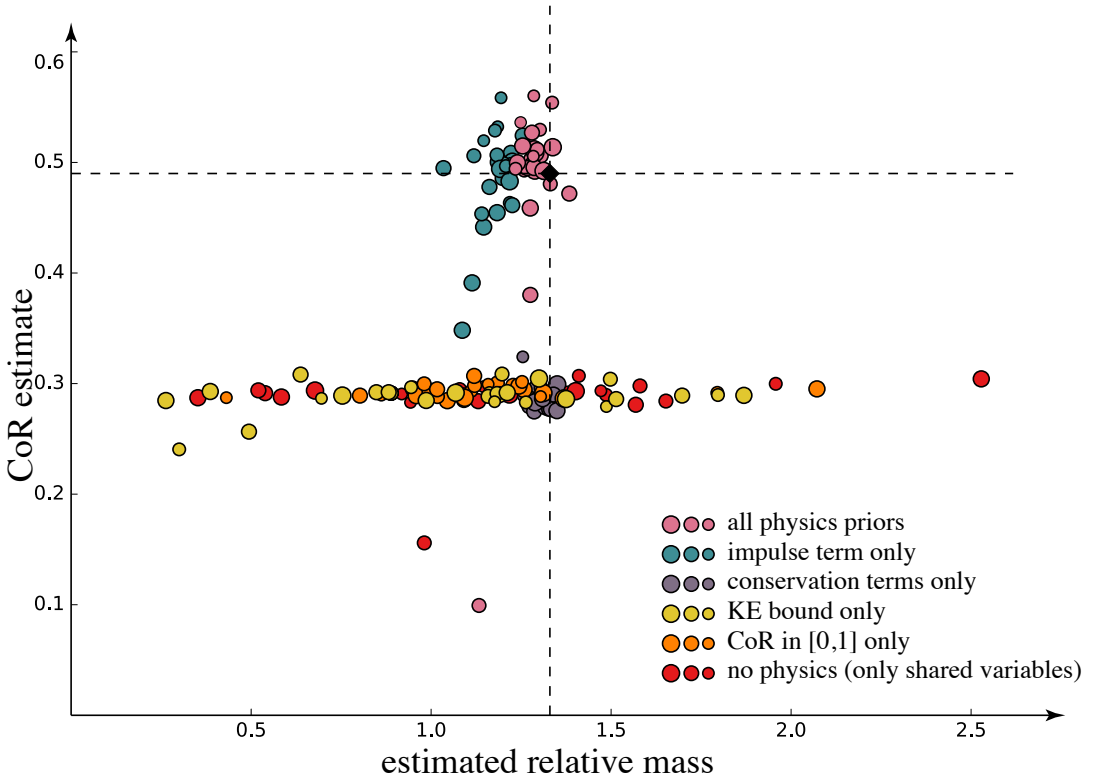


Figure 5.9: Evaluation of the role of different terms. Impulse-based physics coupling is clearly the essential ingredient, while the other terms act as secondary regularizer. Larger circles show experiments with larger noise (positions and orientations). The input was a noise-free run of our automatic parabola initialization and synthetic orientations at least five timesteps away from collision time. Dashed lines denote ground truth, $c = 0.49$ and $m^{b,a} = 1.33$.

solver computes reliable estimates. In all the other experiments, we used 2 orientation annotations for each of the 4 parabolas for a pair of colliding objects.

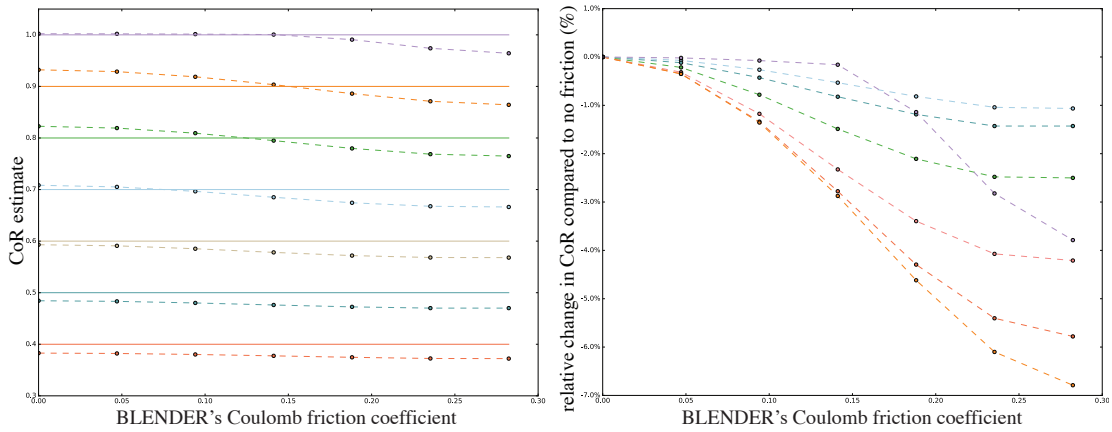


Figure 5.10: Evaluation of the role of friction. We changed the friction coefficient and coefficient of restitution in the input simulation, and ran our algorithm on the simulator's output log (no noise). The c value estimates decrease a bit, which is expected, since some energy is lost to friction. Solid lines on left denote ground truth.

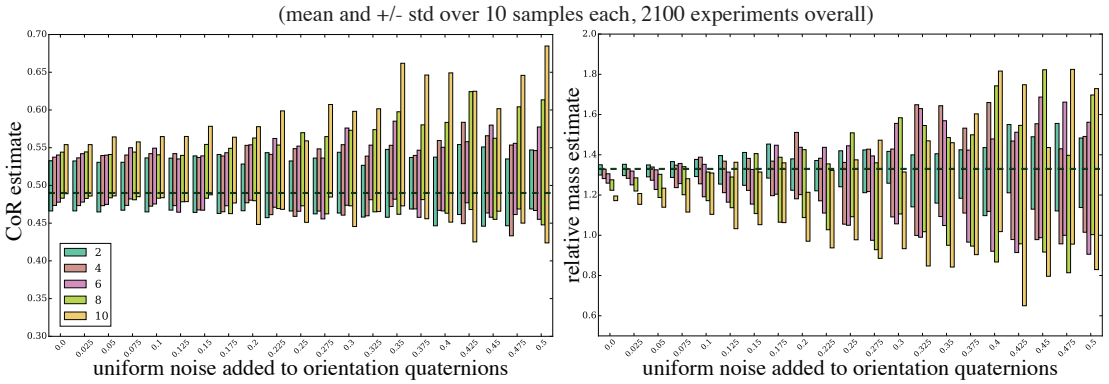


Figure 5.11: Synthetic evaluation of robustness towards noise in the orientations. Gap (2, 4, 6, 8, 10) denotes the distance in time to the closest orientation around collision time. Dashed lines denote ground truth, $c = 0.49$ and $m^{\mathrm{b,a}} = 1.33$.

(iv) Effect of friction. Recall that in our formulation, we ignored the effect of friction. Hence, we evaluated the effect of ignoring friction on our estimates. In the experiments, we changed the (static) friction coefficient and coefficient of restitution of the simulator and re-ran the simulation. We then estimated c and $m^{\mathrm{b,a}}$ using our algorithm on the simulation output. As shown in Figure 5.10, we observed that while the relative mass estimates continued to be good, the coefficient of restitution was underestimated. This is explained by our model ignoring energy loss due to friction, especially when the friction coefficient is (synthetically) increased.

5.4.2 Testing on Real Sequences

We captured the real-world sequences with 1280×720 video with 120fps or 240fps using a regular iPhone6 with known camera intrinsics. The colliding objects were scanned in static positions using KinectFusion.

Potential energy comparison. One way to measure the coefficient of restitution is based on the ratio of energy loss during a collision. However, this requires a very controlled setting, where we can measure the change of energy using only the objects' potential energy, *i.e.,* when they have zero kinetic energy. We use different balls dropped from a controlled height h, measuring the apex of their trajectories after they bounce off the floor. When no energy is transferred into an angular motion, we can compute $c = \sqrt{h_{\mathrm{after}}/h_{\mathrm{before}}}$. We experimented using four different balls (ping-pong, tennis, football, and rugby) and computed the resulting c values. We also used our algorithm to independently estimate c. As this setup involves one static object (the floor), we remove

all unknowns of the second object from our solve. We took 3 sets of measurements for each object. Our estimates $\tilde{c}$ and the values measured with potential energy loss c_m are listed in the table in Figure 5.12. Note that for the examples without any spin, the algorithm ran in an automatic mode as we do not require any orientation annotation.

Our method continues to consistently estimate c even when some energy is stored/converted to kinetic energy throughout the recording, *i.e.,* the ball is spinning, or it starts off spinning (Figure 5.13). The simple estimation with potential energy is not valid for this case, while our formulation is still able to compute an estimate very close to the non-spinning case. Orientation annotations were provided for the scenes *rug-1* and *rug-2*. Please refer to the paper's supplementary video[2] for comparison.

5.4.3 Real-world Cases

Collisions between box-like object pairs. We next focus on collisions between pairs of freely moving objects. In contrast to the potential energy cases, there is no way to measure a ground truth value c_m and we only present the value calculated with our method. We can however measure ground truth for the objects' mass ratio.

[2] `http://youtu.be/rCZ-1yWJP2Q`



scene	$\tilde{c}$	c_m
pong-0	0.55	
pong-1	0.57	0.54
pong-2	0.55	
tennis-0	0.71	
tennis-1	0.73	0.76
tennis-2	0.71	
football-0	0.66	
football-1	0.67	0.77
football-2	0.67	
rug-0	0.80	0.82
rug-1 (spin)	0.80	-
rug-2 (spin)	0.78	-

Figure 5.12: Various c estimates obtained by our method as compared against c_m obtained using alternate potential energy-based estimation, which is valid only when the object has no angular momentum. All the examples are instances of collision of a single object against an infinite mass object (*i.e.,* carpeted ground).



Figure 5.13: Rotating rugby ball scene. Estimated c using our method is $\tilde{c} = 0.78$, matching no-spin measurement 0.8 as detailed in Fig. 5.12.

scene	$\tilde{c}$	$\tilde{m}^{b,a}$	$m_m^{b,a}$
box-1	0.28	0.91	1.0
box-2	0.39	1.21	1.0
box-3	0.20	1.0	1.0

Table 5.3: Results of different collisions of cardboard boxes.

Table 5.3 lists our estimates across three captured videos of the boxes colliding in different situations. Note that in all these examples, the video sequences around (estimated) collision times are ignored by our method as those frames did not provide any reliable position/orientation estimates due to overlapping objects. While the mass ratios are stable, and especially close to the ground truth for the last run, the $\tilde{c}$ vary more strongly. We attribute this partially to the fact that the cardboard boxes exhibit different stiffnesses depending on where they collide (based on the internal structure of the glued paper), in addition to the noise of the centroids estimated from the video. Figure 5.14(a)-(c) shows some frames from the box sequences.

Effect of direct keyframe-interpolation. We evaluate the effect of position and orientation interpolation while ignoring the underlying physics of collision. For this test, we used the initial position and orientation estimates obtained in our initialization stage as keyframes. As shown in Figure 5.15-middle, such a direct interpolation without access to physics-based constraints resulted in 'missing' the actual collision, *i.e.,* the duck and

[3] `http://youtu.be/rCZ-1yWJP2Q`

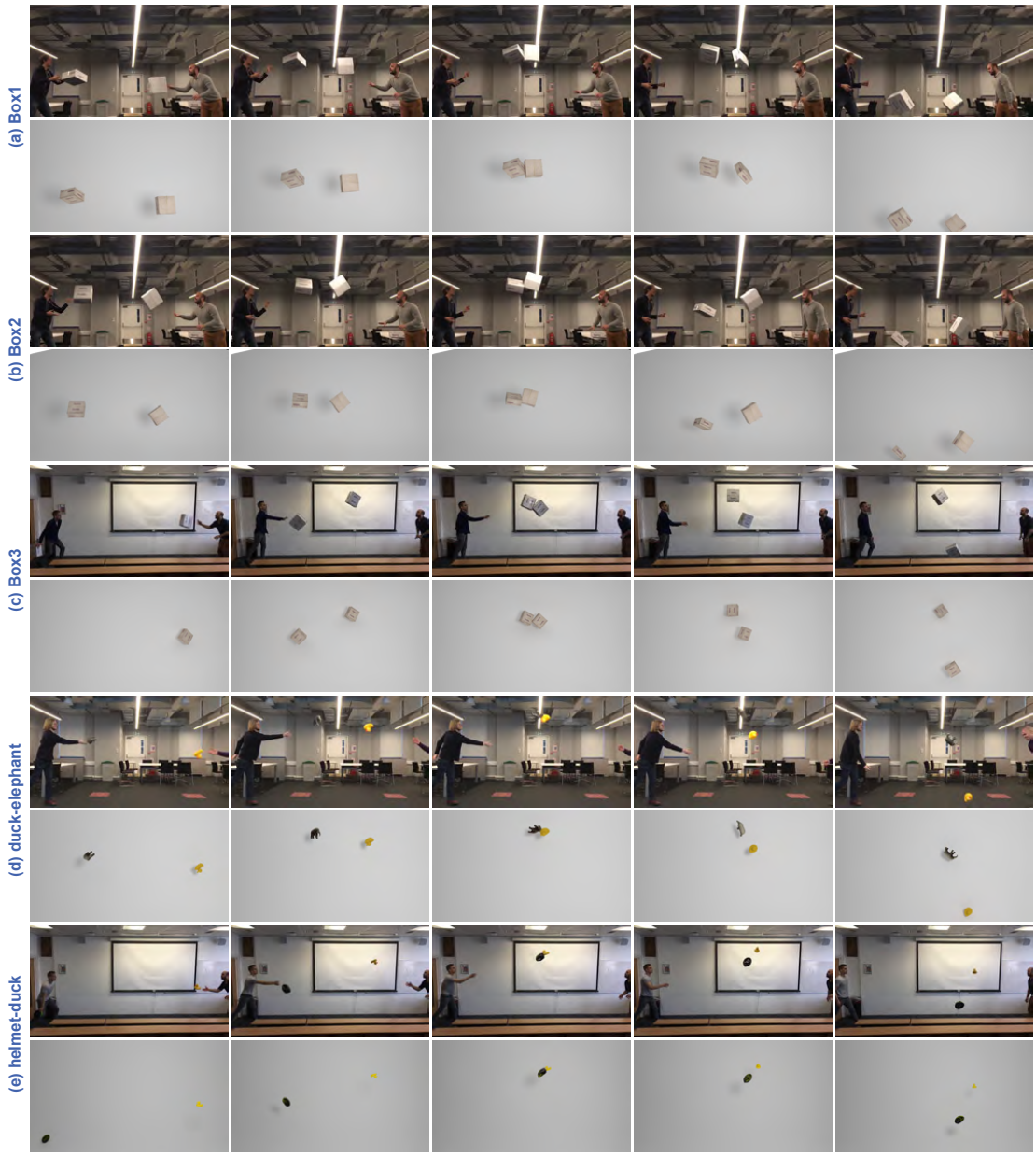


Figure 5.14: We tested our algorithm on multiple collision sequences. Here we show a selection of images of input frames and corresponding reconstructed frames. Please refer to the paper's supplementary video[3] .

elephant 'separated' without actually colliding. In comparison, Figure 5.15-bottom shows that our method accurately reconstructs the collision.

Collisions between non-box objects. We next evaluate how our algorithm performs on collision videos involving non-box objects. We present three sequences of varying complexity: helmet-duck, duck-elephant, and duck-rugby. Table 5.4 lists estimation results.

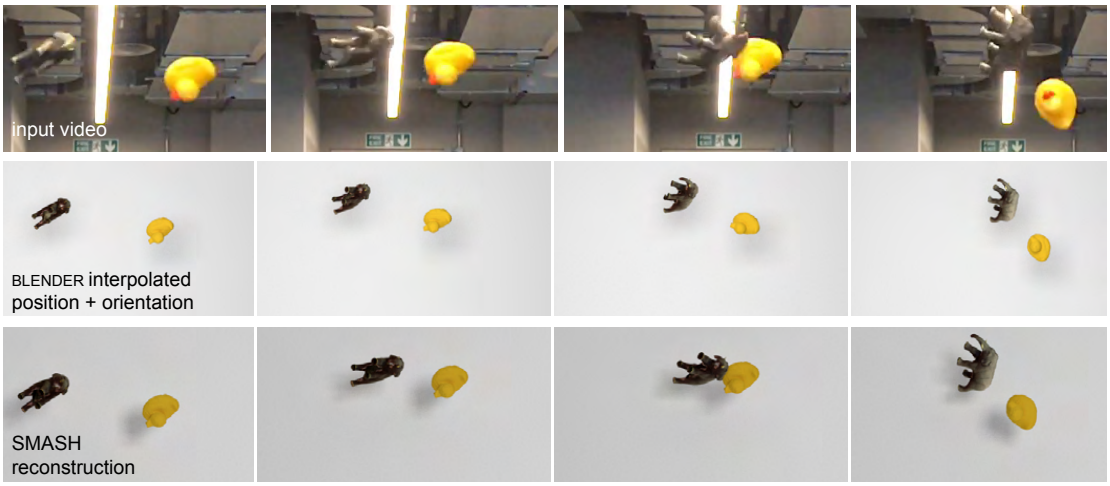


Figure 5.15: Direct keyframe interpolation (middle) versus our reconstructed result (bottom). Note that the interpolated sequences completely miss the collision (middle row, third from left) as it remains oblivious to any physics considerations.

Figure 5.14(d)-(e) show the first two results. As the frame-level comparison shows the reconstructed motion, both in terms of position and orientation, is visually very accurate. Although we validated the estimated mass ratios, we could not validate the accuracy of the estimated $\tilde{c}$ values. Qualitatively the $\tilde{c}$ estimates seem realistic: for example, the bike helmet gives a low estimate as it acts as a shock absorber; in contrast, the elephant and the rugby are bouncy and give high $\tilde{c}$ estimates, with the rugby giving the highest value.

scene	$\tilde{c}$	$\tilde{m}^{\mathrm{b,a}}$	$m_m^{\mathrm{b,a}}$
helmet-duck	0.17	0.81	1.0
duck-elephant	0.46	2.33	2.0
duck-rugby (Fig. 5.1)	0.72	1.43	1.65

Table 5.4: Estimates of internal physics properties ($\tilde{c}$ and $\tilde{m}^{\mathrm{b,a}}$) of complex objects. The helmet clearly dissipates the most energy, whilst a sports ball is the most elastic by design.

The duck-rugby sequence as shown in Figure 5.1 is the most challenging example as the objects remain hidden behind the curtain in the video for a large time interval. Still the reconstruction results closely follow the recorded video. Note that this example is particularly challenging since the duck is comparatively small and featureless, making the initial orientation estimates noisy.

[4] `http://youtu.be/rCZ-1yWJP2Q`

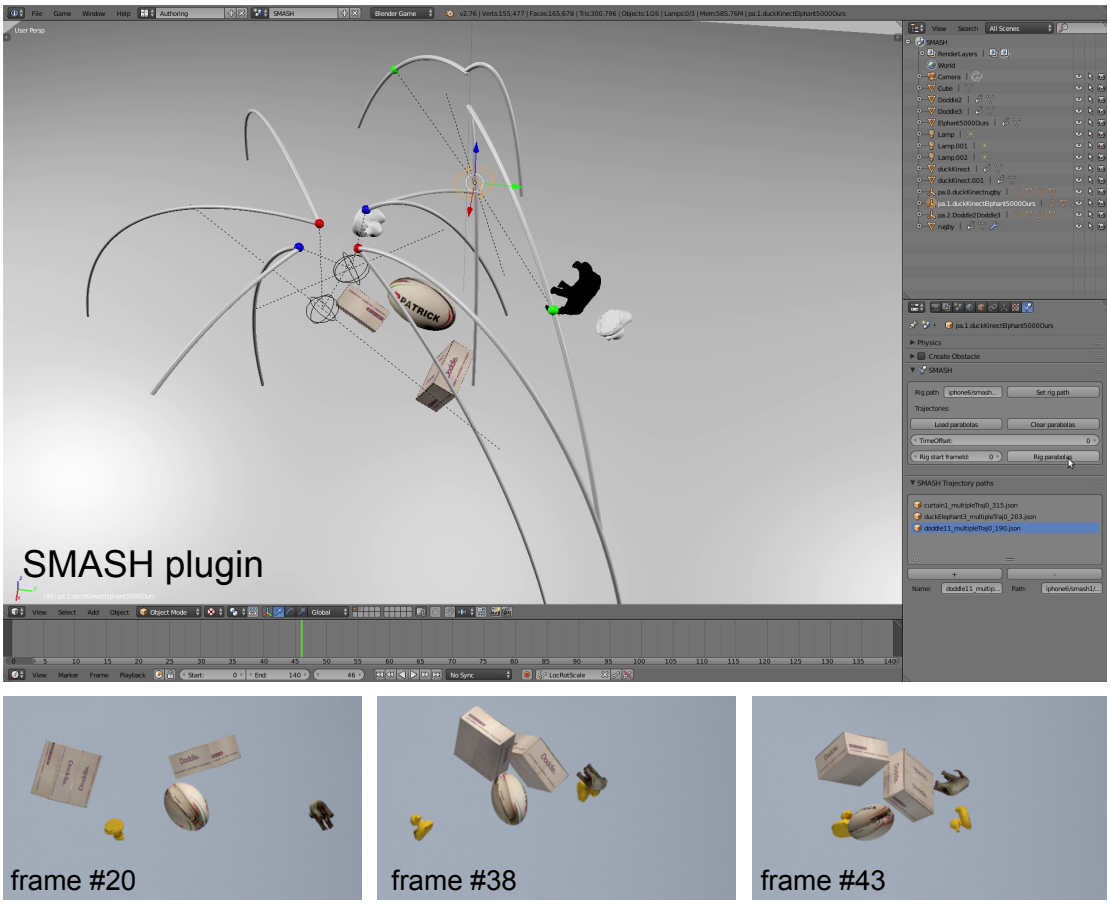


Figure 5.16: Our SMASH plugin (top) can be used to stage and synchronize collisions using the recovered parameters from our reconstruction. Trajectories, mass ratios and c values are read in from the optimization automatically to produce a realistic collision sequence (bottom). Please refer to the paper's supplementary video[4] .

5.4.4 Staging collisions

We developed a BLENDER plugin (see Figure 5.16) to help stage complex novel collision sequences using our parameter estimates from real collisions. The plugin allows the user to combine several observed collisions, time them to synchronize, or even stagger the interactions (*i.e.,* collisions). We used BULLET to obtain the object trajectories as accurately as possible using our estimated collision parameters. Note that the user does not have to guess physical parameters for objects (*i.e., c* and $m^{b,a}$ values). In Figure 5.16 we show the construction of a complex case of three pairs of rigid bodies colliding in mid air and two of them colliding again afterwards on their downward trajectories. Based on our reconstructed solutions, the new simulations behave naturally, and the objects collide as we would expect them to do from their real-world counterparts.

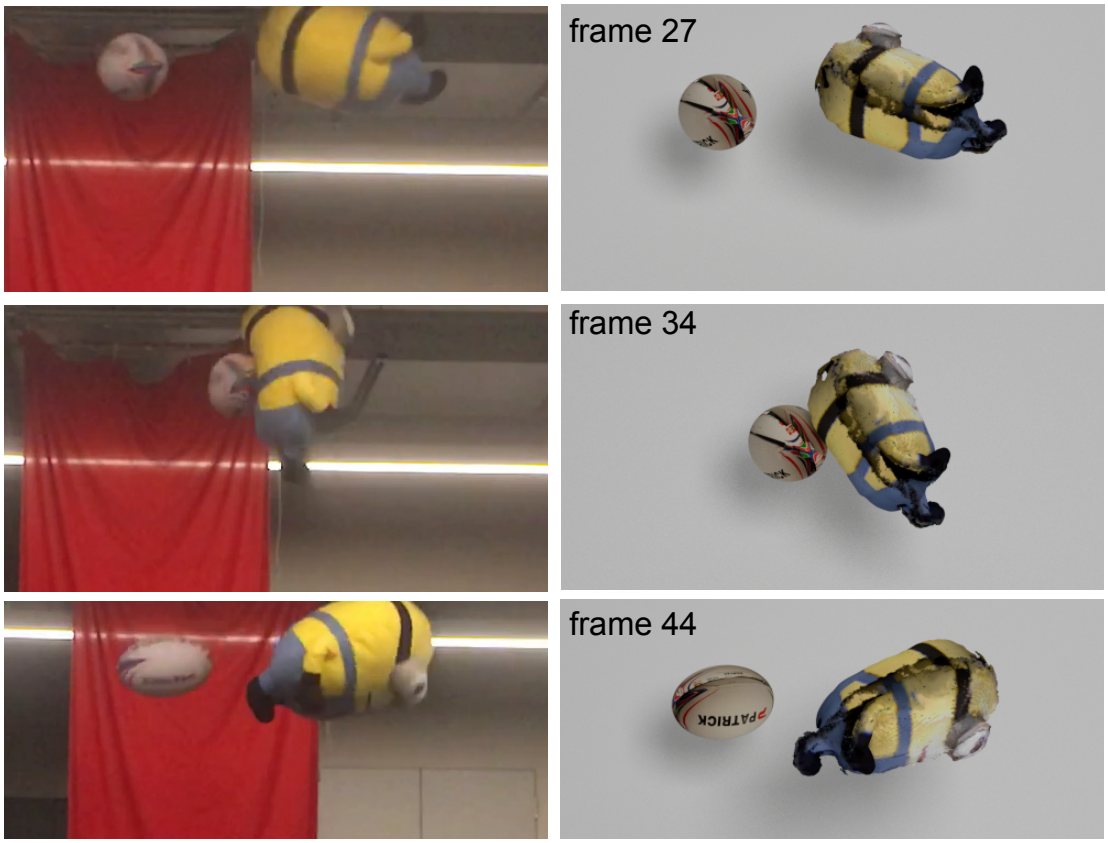


Figure 5.17: Although designed for rigid collisions, we observed that our method still can produce reasonable estimates for near rigid collisions as shown by the minion-rugby sequence. However, the reconstruction is inaccurate around the collision time (frame #34).

5.4.5 Limitations and Discussion

A key limitation of our method is that it only applies to rigid body collisions. In case of deformable bodies, the physical models we employ can be unsuitable. Empirically, however, we observed that our estimates are still reasonable for near rigid collisions (see Figure 5.17). Our algorithm in its current form can reconstruct real-world effects under the assumption of zero friction, but including a friction model in our optimization could potentially increase the accuracy of our reconstructions.

While our current pipeline still requires user input (2 orientations per parabola segment), it is worth pointing out that the user-annotated orientations represent only a small part of the overall input data. All the 3D positions of the objects are reconstructed fully automatically. The user input is typically only required to refine the orientations, which are important for angular motions, and thus help to retrieve the physical parameters.

Also, as we only rely on a single input video, we need enough visual information about the collision event to be visible. For example, if a collision is happening purely

along a view direction ray, our method will most likely underestimate the objects' velocities. Likewise, objects that move very quickly might not exhibit enough change of direction from gravity. In such a case, our method requires more accurate centroids to reliably recover a depth estimate from gravity.

Lastly, it is noticeable in some reconstructions (such as Figure 5.14a) that the objects intersect each other at collision time. This can happen, as our optimization does not include any explicit collision detection or resolution terms. While this might look like a limitation at first, we believe the accurate contacts of the other scenes actually point to a very high accuracy of the reconstructions, which was achieved without taking the exact shapes of the objects into account.

5.5 Discussion

In this chapter, I presented SMASH, a data-driven framework for capturing, reconstructing, and staging collision sequences. The key idea is to regularize the problem of reconstructing collisions of object pairs from videos using laws of rigid body physics. I demonstrate how to reconstruct consistent collision sequences in 3D by observing objects in motion away from the collision instant. The method outputs physical parameters that would otherwise be very difficult to acquire, *e.g.,* the coefficient of restitution and the mass ratio of the participating objects. The information can then be readily used to stage new collision sequences.

While the presented is merely a first workflow for reconstructing collision sequences, many avenues for improvements remain. The assumptions, that the objects interacting have to be in free-fall, *i.e.,* no external forces except gravity are accounted for can be considered limiting for real world scenarios. Natural next steps are to extend the method to rolling motion where surface shape and friction has to be taken into account. Furthermore, the current implementation still expects user guidance to disambiguate the objects' orientations, as despite knowing the 3D shape and appearance of an object in beforehand, the accuracy of differentiable rendering methods limit the accuracy of SMASH's estimates.

Deep learning techniques have become so powerful in recent years, that the natural direction to overcome the above limitations is to train neural networks to infer from monocular video input the external physical states of observed objects interacting

with their environments. In [31], we generated thousands of scenarios of rolling and colliding balls on various geometric and irregularly shaped surfaces. Letting a carefully designed neural network observe these scenarios lead to a system, that is capable to learn an intuition, an implicit representation of how such physical scenarios work.

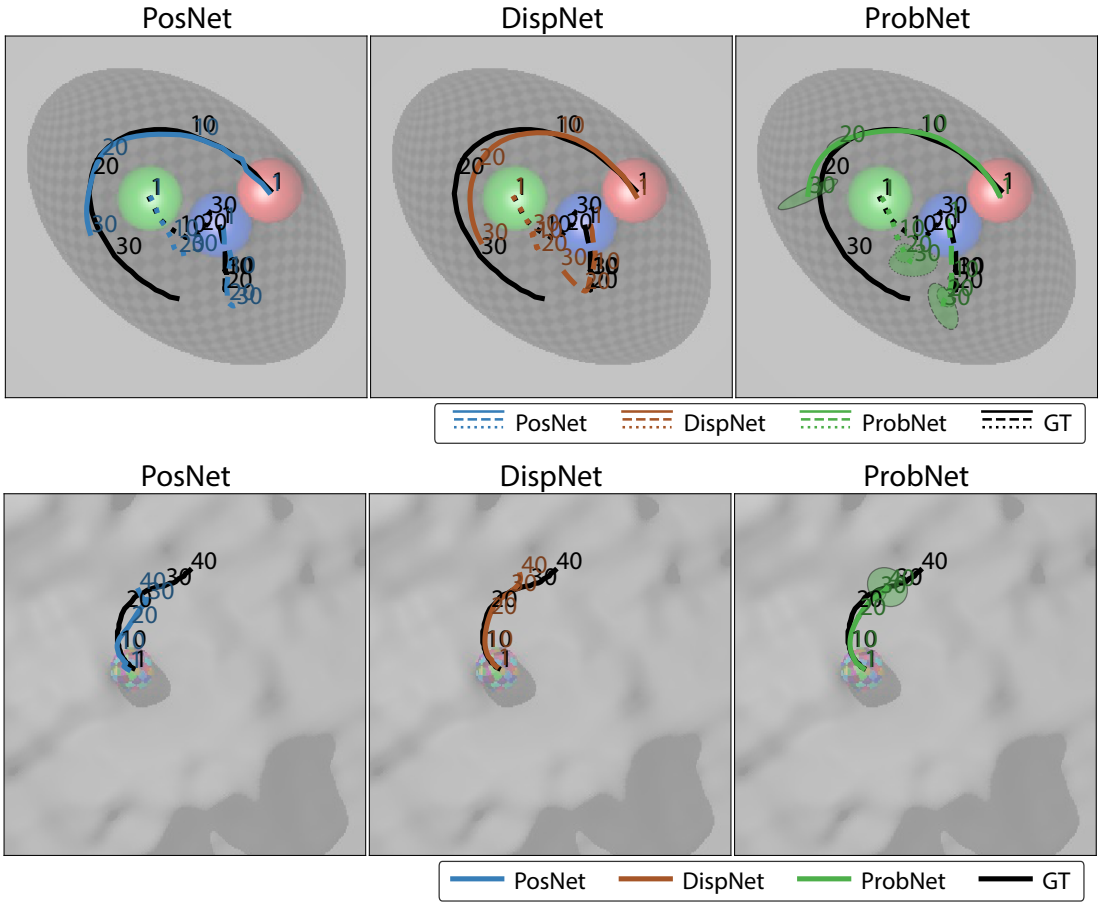


Figure 5.18: Neural motion prediction from first four frames of a video using supervised training. The networks interpret the appearance of the environment and predict the future states of the objects at distant times in the future ($7.5x \ldots 10x$ extrapolation). PosNet outputs absolute screen space positions, DispNet and ProbNet relative displacements compared to detected first positions. ProbNet performs best in position prediction because it also has the opportunity to output uncertainty of its estimates as additional output. Please see the paper's supplementary videos[5] for more examples.

Figure 5.18 shows example outputs of three different recurrent networks designed for the aforementioned purpose. The input setup at test time is always the same, given the first four frames of a monocular video sequence, the system has to predict the position and orientation of one or more textured balls for the next 30 or 40 frames. PosNet was trained to predict absolute position, DispNet to predict relative position and ProbNet to in

[5] `http://youtu.be/bonx5RE1K2Y`, `http://youtu.be/TN5AFsZOLWs`

addition predict the uncertainty of the relative displacement. In all three cases, including the task of predicting the object orientation improves the performance of predicting position. The two main limitation of these networks are: i) they are not performing very well on predicting the object orientation, and ii) they only work on synthetic data, where ground truth external physical parameters are available as supervisory signals. The latter means, that only such physical scenarios can be considered, where physical simulators exist and that we have to trust the accuracy of these simulators. Both above limitations mean, that further improvements are necessary before the networks can be prepended to SMASH's pipeline. In order to address the latter limitation of the networks in [31], in [30] the first unsupervised predictor of physical states was created.

The key contribution is, that it doesn't need any ground truth information about physical states for training anymore granting follow-up works the opportunity to extend the problem domain to physical processes that are difficult to model computationally. The key idea behind the solution is to exploit the fact, that the evolution of physical systems can be described by well-understood laws that are strongly dependent on time as a changing variable. This dependence together with the fact, that the input to our problem is a series of images strictly ordered in time, a video, allows us to create a special form of self-supervision.

The key finding of [30] is that when one appends to a recurrent neural network a classifier that has to solve an auxiliary task of deciding, whether a certain short sequence of input images was ordered in time or not, the loss of this classification task propagated through the classifier can train the recurrent part of the network to learn to focus on the dynamic parts of the scene. Focusing on the dynamic objects will allow it to successfully develop an internal representation of realistic physics, which at training time allows it to answer the question, whether the input sequence was scrambled in time or not, and at test time the intermediate representation of where it thinks the object(s) are in the images can be plugged in to the previous system [31] to then perform physically realistic motion extrapolation. The method composed in such a way never had to have any supervision in the form of explicitly description of the environment's physical properties, and thus may be extendable to more complicated scenarios where simulators are computationally impractical. Although it is very promising to see, that the neural network seemingly learned to take cues from the visual appearance of the input images to decide the future



Figure 5.19: Motion prediction from first four frames of a video trained without supervision purely from real videos. The networks are capable of interpreting the appearance of the environment and the apparent initial displacement of the objects to output the position of the objects far in the future in time. In the last row, the ground truth motion was a collision, whilst in the output predictions, the objects miss each other and continue straight. In the heightfield cases, the underlying surface is so uneven, that even a small misestimation of position results in large long-term deviation of trajectories.

trajectory of the objects, it is on follow-up work to prove, that there is a true separation of objects and environment in the internal representation. Preliminary results are under review [32][6].

Another main limitation of the above methods is, that they assume a perfectly static environment and camera. Even minor movements of the background sometimes barely visible to the human eye did manage to prevent the networks from being able to identify the "physically salient" parts of the input footage.

In summary, the emergence of efficient methods to training deep neural networks represent promising opportunities to continue to explore ways to add prior information about our physical environment to our toolbox when performing scene analysis.

Finally, I believe SMASH and follow-up works only represent a small step towards a tighter integration of video analysis, dynamic geometry acquisition, and physics simulations. There are many other physical phenomena, such as deformable objects or fluids, that could be included in order to on one hand better understand what is happening in a recorded event, and on the other hand reconstruct the underlying physics for computer animation purposes.

[6] https://arxiv.org/abs/1905.10793

Chapter 6

Human interactions as non-visual cue

6.1 Introduction

Computational understanding of monocular videos that capture human-object interactions in physical spaces is critical for many emerging fields such as virtual and augmented reality, smart home systems, assisted living, and robotics. Such applications require access to object arrangements embedded in the physical spaces along with the common human-object interactions performed in such spaces. For example, our future personal robot assistants should know our working habits along with the supporting object arrangements in our living rooms or workspaces. Hence, a joint understanding of scenes and human actions from the input feed is necessary.

While both scene understanding and human performance analysis are highly popular research areas, traditionally, researchers have tackled them as two separate problems. On the one hand, *scene estimation methods* such as KinectFusion [101] and Bundle Fusion [24] can produce high-quality static indoor reconstructions, while the likes of DynamicFusion [113] can capture non-rigidly deforming scenes by fusing depth information across space and time. These methods, however, require the sensor to be manually moved to see around occlusions making the capture process cumbersome. On the other hand, *human performance capture methods* either use multiple sensors [167] or monocular video [104, 130] but assume performances to be free from object-induced occlusions. However, in monocular footage that capture human actions in physical spaces, objects and human motions regularly *occlude* each other during interactions.

[1] `http://geometry.cs.ucl.ac.uk/projects/2019/imapper/paper_docs/MonszpartEtAl_iMapper_Siggraph_2019.mp4`

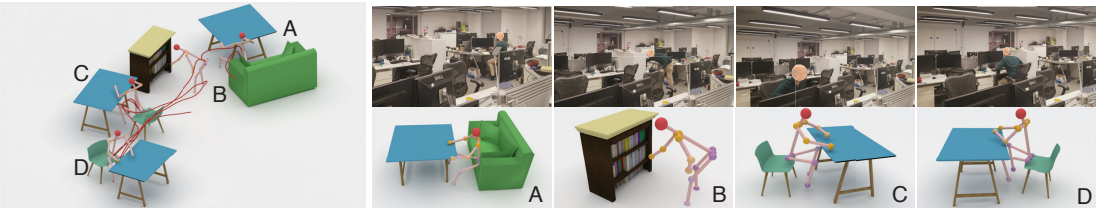


Figure 6.1: Interaction-guided scene mapping. We present iMAPPER that discovers potential human-object *interactions* in an input monocular video and utilizes them to infer the object layout of the recorded scene containing medium to heavy occlusion. We show the final generated 3D scene as well as recovered interactions (Scene13). Note that although *only* the 2D human joint detection (left) is available to the algorithm, here we additionally show reference video frames (corresponding to A, B, C) to help judge the original scene layout. Please refer to the paper's supplementary video[1].

Filling in missing information due to such occlusions is ambiguous due to the diversity of possible scene configurations and is handled poorly by current methods.

While indoor scene configurations can be extremely rich and diverse, we observe that many of them are linked by a common thread — *they are regularly inhabited by humans*. Moreover, in similar scene configurations, humans tend to perform similar actions (*c.f.,* [85]). Examples of such actions include sitting on sofas, picking up books from shelves, or walking around obstacles. Instead of tackling scene estimation and human performance capture separately, we propose to exploit the captured human performance to better infer consistent scene layouts.

A fundamental challenge in reaching the above goal using monocular video in natural surroundings is *occlusion* arising out of human-object interactions. A successful solution needs to tackle two problems: first, hallucinating information about partially or fully hidden objects; and second, recovering from noisy human performance estimates from monocular videos, especially in regions of medium to high occlusion. It is believed that, as humans, we focus on the interactions of the actors with the objects in a scene (referred to as 'anticipation' in [112]), instead of separately identifying objects and human performances. Detecting such interactions helps compensate for missing information in *both* objects and performances. For example, in the video for the scene shown in Figure 1, we can 'see' the person walking behind the desk and sitting down; from that, we can imagine both the person's sitting pose over time and the location of the unseen chair/sofa. Similarly, for the person picking up an object from the shelf.

In this chapter we propose iMAPPER, a data-driven method, that accomplishes a similar feat by *utilizing human motions to infer object placements*. As a data-prior, we leverage a database of interactions between humans and local objects over time, which we call *scenelets* (extracted from the PiGraph dataset [139]). Our key observation is that state-of-the-art methods (see Section 6.7) are now reliable for detecting visible parts of the human performance, and hence the local objects being interacted with, even if partially or fully occluded, in the scenelets matched to such human performance detections can be used to provide good candidate object layouts associated with such detected human-object interactions.

Starting from a monocular video, we use a state-of-the-art human pose detector to identify initial joint estimates over time and analyze them to identify 'informative' space time snapshots representing potentially informative interactions. We then utilize the snapshots to retrieve matching local scenelets and solve a global optimization to extract a consistent subset of these scenelets, arrange them, and inherit their associated objects and human actions to produce both static objects and a human performance that are mutually consistent, and agree with the input video.

We extensively evaluated iMAPPER on a range of scenes of varying complexity. Our quantitative evaluation against manually annotated ground truth data, and qualitative evaluation through a user study demonstrate that iMAPPER produces realistic and consistent object layout and human-object interaction estimation even for scenes with significant amounts of occlusion.

In summary, our main contributions are:

(i) proposing the first method that discovers and utilizes human-object interactions to produce a 3D scene layout from a monocular video showing human interactions in natural settings;
(ii) extracting informative space-time human action snapshots and matching them to a scenelet database; and
(iii) combining the matched candidate scenelets into a consistent 3D scene layout and human performance by a novel global optimization, and evaluating the proposed iMAPPER method on a range of challenging examples.

6.2 Scenelets: Representation and Descriptors

iMAPPER heavily relies on identifying and utilizing human-object interactions for scene layout and mapping. We start by describing how we represent the space of possible interactions as a database of 'scenelets', introduce suitable descriptors to query into this database, and the important notion of *interaction-saliency* to identify informative scenelets with strongly correlated human-object interactions before presenting the iMAPPER algorithm in the next section.

As our human-object interaction database, we use the PiGraphs dataset [139] that contains a set of scenes captured by a commodity depth sensor, each containing a human performance and a set of associated labeled objects (*e.g.,* tables, sofas, chairs and bookshelves). From this dataset, we extract short sequences representing interactions between the human actor and scene objects. We call such short sequences of frames *scenelets*.

6.2.1 Scenelet Representation

Each scenelet $\mathscr{S}_l = \{\{s^l_{k,t}\}, O^l\}$ consists of a short motion clip with known 3D joint positions and a set of static objects. We denote with $s^l_{k,t}$ the location of skeleton joint k in frame t of scenelet l. Objects $O^l = \{o^l_1, \ldots o^l_n\}$ of scenelet l are defined by a placement p, a rough approximation of their geometry κ, and a class label b; *i.e.,* each object is encoded as a triplet $o = (p, \kappa, b)$. We assume objects can only rotate around the up direction, leaving four degrees of freedom for the placement of an object: $p = (x, z, \theta)$, where x and z are the location, and θ the orientation of the object. Similar to the original dataset we approximate the geometry of objects by unions of cuboids, and the label b^l_i describes the object type (*e.g.,* chair, table, bookshelf) from a predefined set of categories. Both the motion clip and the objects are stored in the local coordinate frame of the scenelet defined by the pelvis location and the forward-facing direction of the skeleton in the center frame of the motion clip. Figure 6.3 shows some example scenelets.

6.2.1.1 Scenelet parameterization

When constructing scenelets, we make a design choice regarding the time-duration of the motion clip used for each scenelet based on two factors. First, the speed at which an interaction is performed should not affect the contents of a scenelet. For example, if a scenelet captures a fast 'sitting-down' performance then a slower version of the same

interaction should also be captured by a single scenelet. This property is necessary to ensure that interactions captured by scenelets are comparable. Second, we assume that interactions are local in space, *i.e.,* the actor does not traverse a large distance during the interaction.

We select the duration of scenelets such that the skeleton (*i.e.,* the pelvis joint) traverses a constant arc length as this satisfies both time invariance and locality requirements. For increased robustness, we smooth the pelvis trajectory when computing the arc length with 10 iterations of a moving average. We set the spatial extent of the smoothing kernel to an arc length radius of 1 cm (the input skeletons were recorded at 30 Hz).

6.2.1.2 Scenelet construction

The PiGraphs dataset describes human performance with a set of 16 joint locations per frame. We start by sampling the center of each scenelet's motion clip at regularly spaced intervals on the arc length of the pelvis joint's trajectory. The start and end of the motion clip in each scenelet is then defined as this center point $\pm$ half the scenelet arc length. The objects of the scenelet are chosen as the subset of objects within roughly an arm's reach (within 1 m radius when projected to the ground plane) of the actor at any point in the motion clip, *i.e.,* the objects the actor can potentially interact with, manually corrected where needed.

6.2.2 Scenelet Descriptors

In order to compare two scenelets $\mathscr{S}^1, \mathscr{S}^2$ and compute their distance $d(\mathscr{S}^1, \mathscr{S}^2)$, we define two descriptors for each scenelet: a *motion descriptor* Ψ and an *object descriptor* Φ. As explained later in this section, we utilize these descriptors to identify informative scenelets via their *interaction-saliency* scores.

6.2.2.1 Motion descriptor

The *motion descriptor* Ψ captures the human actor motion over the scenelet duration and compactly describes the motion clip of a scenelet with a fixed-length vector as the concatenation of a fixed number of static pose descriptors $\Psi := (\psi_1, \ldots, \psi_t)$, we use $t = 15$ samples. Static pose descriptors are based on 14 robust joint-line distances as proposed by Zhang et al. [188]. The distance between two motion descriptors is defined using a weighted L_2 distance between the corresponding static pose descriptors,

assigning more weight to center frames. The descriptors ψ_i should evenly cover the motion and be invariant to the speed of the motion. Thus, we evenly distribute these samples along the trajectory of the motion clip in a 17D space of the combined pose descriptor and global skeleton location (taken to be the 3D location of the pelvis).

6.2.2.2 Object descriptor

The *object descriptor* Φ encodes statistics of relative object layouts with respect to the actor and is defined as a set of histograms with one histogram per object category. The histograms capture the 2D placement of objects, projected to the ground plane. Our histograms are 5×5 square grids (in a coordinate frame with the center pose facing the forward direction) and each bin Φ_j describes to what extent any object of the same category in the scenelet is located in this bin. We define the value in a bin as the maximum coverage of the bin by any object. To handle both objects smaller and larger than the bin, we normalize by the smaller of either the bin area or the object area as,

$$\Phi_j = \max_i \left\{ A\left(\Lambda(o_i) \cap \phi_j\right) / \min\left(A\left(\Lambda(o_i)\right), A(\phi_j)\right) \right\},$$

where $\Lambda(o_i)$ is the projection of object o_i to the ground plane, ϕ_i is the part of the ground plane covered by bin i, and $A(x)$ is the area of x. Figure 6.2 shows an example of an object descriptor.

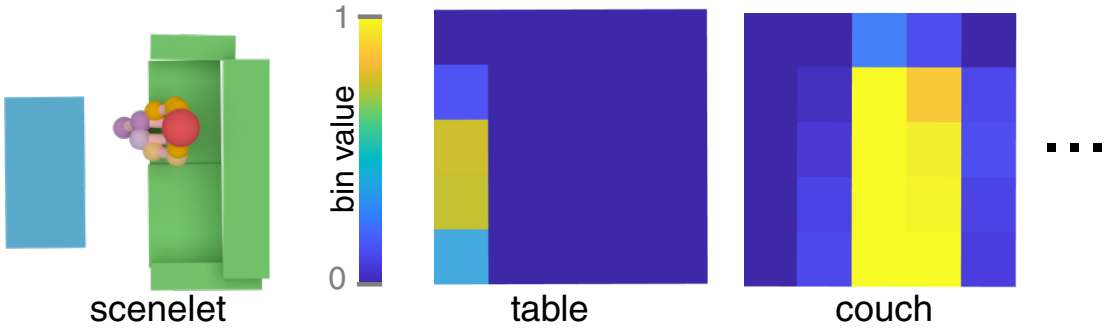


Figure 6.2: Object descriptor Φ. Object descriptors compactly represent the object arrangement of a scenelet. One 5×5 histogram per category stores the layout of objects of this category relative to the scenelet center.

6.2.3 Interaction-saliency for Scenelets

In order to identify informative human-object interactions, we associate each scenelet with an *interaction-saliency* score. Scenelets with high *interaction-saliency* are likely to

contain informative interactions with objects and they have objects within interaction range that are typical or characteristic of the scenelet's motion.

Thus, we define the uniqueness or saliency of a bin in an object descriptor to describe how typical an activation of this bin is for similar motion clips. For example, for a sitting-down motion, a couch or a chair at the center bin of the histogram will have high saliency. The main intuition behind this saliency score is to distinguish between objects that are consistently related to a given interaction, and objects that are near the motion but unrelated to the interaction. In other words, we associate repeated and consistent presence of objects as a reliable 'witness' to the particular interaction represented by the human pose. The bin-saliency of an object descriptor bin is computed as a weighted average of that bin's activation over similar motion clips, where similarity weight is defined with a Gaussian kernel in the space of motion descriptors as

$$h_j^l = \frac{\sum_{k=1}^{m} \Phi_j^k \, \mathscr{G}(d(\Psi_k, \Psi_l)|0,\sigma)\rho_k^{-1}}{\sum_{k=1}^{m} \mathscr{G}(d(\Psi_k, \Psi_l)|0,\sigma)\rho_k^{-1}} \ ,$$

where h_j^l is the saliency of bin j in scenelet l, Φ_j^l is the bin value of scenelet l, $\mathscr{G}$ is a Gaussian kernel taken over the distance d between the motion clip descriptors defined earlier (we set $\sigma = 13$), and ρ_l is the *density* of scenelets at the origin of scenelet l. We measure the density of scenelets as the spatial density of the origins of all scenelets that were obtained from the same scene in order to remove any bias introduced due to multiple scenelets taken from nearby parts of the scene. The summation is computed over all the scenelets in the database and finally, we define the *interaction-saliency* of a scenelet $\mathscr{S}^l$ as the maximum of the bin-saliency, *i.e.,*

$$H^l = \max_j (h_j^l).$$

6.3 Algorithm Overview

The input to iMAPPER[2] is a monocular video showing a person interacting with objects. Our goal is to synthesize a realistic scene layout along with consistent human performance that explains the input video.

[2]Project code and occlusion benchmark dataset at `http://geometry.cs.ucl.ac.uk/projects/2019/imapper`

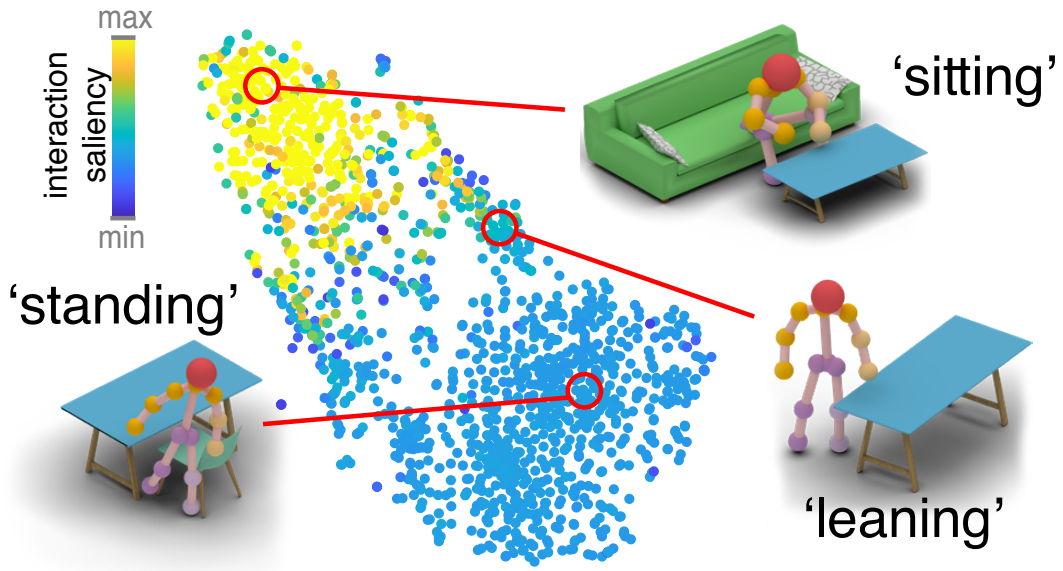


Figure 6.3: Scenelet's interaction-saliency We show a t-SNE embedding of *interaction-saliency* scores of the scenelets in our database. Warmer colors denote scenelets with higher *interaction-saliency*. The close-ups show scenelets with a standing sequence, a sitting sequence, and a leaning sequence. Standing sequences have low *interaction-saliency* because they are not specific to the neighboring objects.

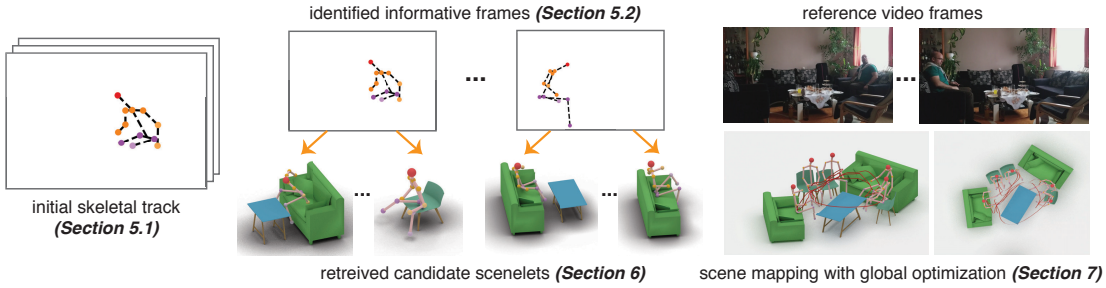


Figure 6.4: Overview Given an input monocular video, we first extract an initial skeletal track with confidence scores (Section 6.4.1). We identify informative frames in the input sequence by assigning each frame an *interaction likelihood score* based on their matching scenelets (Section 6.4.2). For each informative frame, we identify the top matching candidate scenelets (Section 6.5). Finally, we perform a global optimization that selects and places such candidate scenelets to form a consistent 3D scene and human performance (Section 6.6). We show the final synthesized 3D scene, the recovered trajectory of the person (in red), and some of the detected interacting human poses. The reference video frames show that the synthesized 3D scene provides a consistent explanation.

When watching performance of a human actor in a scene, there are several cues that help recover consistent explanations for scene objects and performance sequences, even under partial occlusion. Specifically, both presence and absence of human-object interactions carry valuable hints: on the one hand, detecting a person interacting with objects provides information about the potential types and locations of the surrounding objects. On the other hand, observing a person walking carries information about empty

regions of the scene. iMAPPER explores such cues and proceeds in three main stages, as described next.

(i) *Identifying informative interactions:* In Section 6.4, we identify potentially informative space-time segments of the input skeletal motion as useful human-object interactions. Starting from a monocular video, we first use state-of-the-art human pose detectors [160, 130] to generate an initial human skeleton track over time. Using the notion of *interaction-saliency* defined in Section 6.2, we assign scores to parts of the skeletal motion by matching against database scenelets and assessing how informative the parts are based on the *interaction-saliency* scores of these matched scenelets.

(ii) *Retrieving matched interactions:* In Section 6.5, we formulate an energy function to help retrieve matching scenelets corresponding to the skeletal motion segments extracted in the previous step. During this process we encourage consistency measured based on how well the fitted models explain the presence or absence of skeleton joint detections in each video frame. The matched scenelets, in turn, provide local objects as candidate completions for the occluded parts of the scene.

(iii) *Scene mapping via global optimization:* Finally, in Section 6.6, we solve a global optimization by minimizing an energy that additionally considers plausibility criteria (*e.g.,* path smoothness and intersection avoidance) to obtain a static scene layout and a consistent human performance that matches the input monocular video. Specifically, we formulate a selection problem to extract a subset of scenelets among the matched candidate ones to constitute our synthesized scene, and optimize the placement of these chosen scenelets and skeletons to provide a realistic explanation for the input video.

6.4 Identifying Informative Interactions

Starting from a monocular video, as preprocessing we use existing human pose trackers to form initial skeletal joint tracks over time, and then identify informative interaction segments using a data-driven notion of *interaction-saliency*.

6.4.1 Generating an Initial Skeletal Estimate

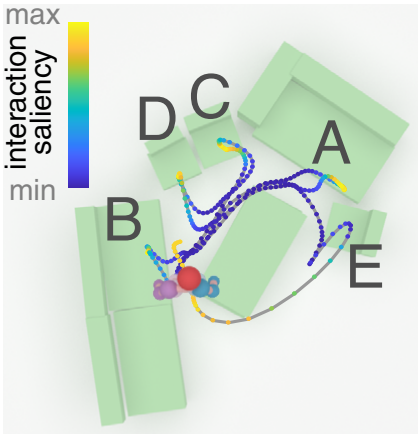
We apply state-of-the-art static pose detectors to obtain initial skeletal estimates from the input monocular video. In each frame, we detect the image-space skeleton of the actor

consisting of n_j joint locations along with local 3D pose estimates (*i.e.,* pelvis is always at the origin). In our experiments, we tested with (i) 2D keypoints detected by CPM [174] and grouped based on the heuristic of Tomè et al. [160] or (ii) LCR-Net++ [130]. Given a video, for each joint k detected in frame t, $u_k^t \in \mathbb{R}^2$ denotes its image-space location and $c_k^t \in [0,1]$ its confidence. While CPM directly provides confidence values, for LCR-Net++ we compute them using pose proposal variance (see Appendix 6.6.3.1). We note that these initial pose estimates are often highly noisy in the presence of occlusions, *e.g.,* around human-object interactions. We next try to match the initial 2D skeletal pose estimates (using a sliding window approach) to database interactions by fitting them close to the initial 3D pelvis path to identify informative space-time segments.

6.4.2 Estimating Interaction-likelihood Score

Our goal is to assign scores to the video frames indicating the likelihood to contain *informative* interactions between the actor and the objects (see Figure 6.3), based on which we can fetch potential scene objects. We start by fitting the scenelets in our database to each video frame and use the *interaction-saliency* (see Section 6.2.3) of the matched scenelets weighted by their matching quality to determine the probability of an interaction.

Specifically, for any frame at time t, say $\{\mathscr{S}^i\}$ be the top $\mathbb{K}_{\text{sal}}$ matched scenelets ($\mathbb{K}_{\text{sal}} = 20$ in our tests) with corresponding *interaction-saliency* given by H^i and matching quality by w_i. This quality w_i measures how well the 3D skeletal joints in frames inside a window of $[t-\tilde{t}, t+\tilde{t}]$ frames (we use ± 10 frames) match the human motion in the scenelet $\mathscr{S}^i$ according to the fitting energy defined in Equation (6.5) . We then define the *interaction-likelihood* score for frame t simply as the weighted average $\sum_i w_i H^i / \sum_i w_i$. The inset figure shows an example scene where the final recovered actor trajectory is color-coded based on the interaction likelihood of the corresponding frames. The regions denoted by letters have higher score indicating sitting interactions.



Our goal is to fit scenelets only to parts of the
video that contain interactions, *i.e.,* are informative of the objects in the scene. Thus, we perform non-maximum suppression of the interaction-likelihood over the video frames. At this stage, we have assigned scores to the frames quantifying how informative they are for assisting in subsequent interaction-based object placement.

6.5 Retrieving matched interactions

We quantify the consistency of alignment of a given 2D skeletal track $\{u_k^t\}$ with associated detection confidence $\{c_k^t\}$ against a scenelet $\mathscr{S}$ over its placement P (see Section 6.2.1) using an energy function that penalizes inconsistency as,

$$L(\{u_k^t, c_k^t\}, \mathscr{S}, P) = w_r L_r + w_o L_o, \tag{6.1}$$

where L_r measures the *reprojection* error between the 2D joints and the projection of 3D joint locations; and L_o penalizes the presence of *occlusions* of skeleton joints in the video that are not explained by occlusions in the scenelet objects. We next describe the terms.

Reprojection term (L_r). The reprojection term is a standard back projection term that penalizes the distance from the image-space 2D joint locations detected in the video to the projected scenelet joints q_k^t as,

$$L_r = \sum_t \sum_k c_k^t \left\| \Pi q_k^t - u_k^t \right\|_2^2, \tag{6.2}$$

where Π is the camera projection matrix (we assume the intrinsic camera parameters to be known) and u_k^t are the detected input 2D joint locations with respective confidence c_k^t.

Occlusion term (L_o). The occlusion term is a novel term that encourages consistency between joint occlusions observed in the video and occlusions of joints induced by scenelet objects as seen from the camera. In other words, we require the synthesized objects to explain observed occlusions. This cost is asymmetric, *i.e.,* missing joint detections occur either due to false negatives in joint detections, or due to occluding objects. The reverse is, however, *not* true: the joint detector may, in some cases, also predict the position of occluded joints with high confidence. We define an asymmetric

occlusion error as,

$$L_o = \sum_t \sum_k F(v(q_k^t, O, \Pi), c_k^t), \tag{6.3}$$

where $v(q_k^t, O, \Pi)$ denotes the visibility of joint q_k^t given the scene objects O and the current camera information Π. In order to have non-zero gradients that are necessary for gradient-based solver, we define v as the signed distance of joint q_k^t to the *occlusion volume* induced by O that is the volume that remains invisible from the camera. We then define the asymmetric occlusion error F for a joint and a set of objects is as,

$$F(v,c) = \begin{cases} (c-0.5)^2 v^2 & \text{if } c-0.5 < 0 \text{ and } v > 0 \\ 0 & \text{otherwise,} \end{cases} \tag{6.4}$$

where $c \in [0,1]$. Note that this function is non-zero *only* when low-confidence joint detections are explained by visible joints. Finally, we can quantify the alignment error between given 2D skeletal tracks against a scenelet $\mathscr{S}$ as,

$$L^\star(\{u_k^t, c_k^t\}, \mathscr{S}) = \min_P L(\{u_k^t, c_k^t\}, \mathscr{S}, P). \tag{6.5}$$

We solve Equation (6.5) by a gradient-based optimization and retrieve the top $\mathbb{K}_{\text{glob}} = 5$ matching scenelets. For efficiency, we only consider the best $\mathbb{K}_{\text{loc}} = 200$ scenelets from Section 6.4.2 in this section. Next, we describe how to globally select among these retrieved scenelets and obtain a final scene layout.

6.6 Scene Mapping via Global Optimization

Our goal is to synthesize a scene consisting of 3D joint locations $q_k^t \in \mathbb{R}^3$ for each video frame, describing the human performance, and a set of objects $O = \{o_1, \ldots o_{n_o}\}$. The 3D joint locations at each frame are obtained from either one of the matched candidate scenelets, or fitting a 3D skeleton to the 2D joint detections in the video; while objects are only taken from the selected candidate scenelets. Which of the two models we fit to a given part of the video depends on two factors: the estimated amount of joint occlusion observed in the video (*i.e.,* the confidence of the joint detection signal) and the estimated probability of object interactions.

First, in segments of the video with high interaction-likelihood, our task is to pick a scenelet among the top $\mathbb{K}_{\text{glob}}$ matches obtained previously, *i.e.,* we have to solve a selection problem. Second, in segments of the video with low interaction-likelihood, matched scenelets are less useful. Instead, we match the initial local 3D human pose estimates to the image-space joint detections. In the following, we describe these two scenarios, and then define a global energy that when minimized produces the final solution.

6.6.1 Video Segments with High Interaction-likelihood

Here, we pick among the matched scenelets to both populate the scene with objects involved in interactions and explain occlusions of joints due to these objects. Thus, joint occlusions can help in both choosing and placing the scenelets. For a given video sequence, we would like to choose and place a scenelet such that the objects of the scenelet explain the joint occlusions observed in the video sequence.

We start by modeling the assignment of scenelets in our dataset to time intervals in the video. Given a video with n_v frames and a dataset with m scenelets, only a single scenelet can start at any frame of the video. The (unknown) scenelet assignment can therefore be expressed with a binary matrix $\mathscr{X} \in \{0,1\}^{m \times n_v}$ with

$$\mathscr{X}_{lt} = \begin{cases} 1 & \text{if scenelet } l \text{ starts at video frame } t \\ 0 & \text{otherwise.} \end{cases}$$

The constraint that scenelets should not overlap in time can be formulated as,

$$\eta_t = \sum_{l} \sum_{i=1}^{\max(t,n_l)} \mathscr{X}_{l(1+t-i)} \leq 1, \;\; t = 1 \ldots n_v,$$

where n_l is the number of frames in scenelet l and η_t measures the integer number of scenelets that overlap with frame t and thus needs to be less than or equal to 1. Since only a single scenelet can start at any frame t, we model scenelet placement with one set of parameters per frame $P = \{P_1 \ldots P_{n_v}\}$, where $P_t = (x, y, z, \theta)$ is the placement of the scenelet starting at t, with x, y, and z the location and θ the orientation of the scenelet. The 3D joint locations $\hat{q}_k^t$ in video sequences covered by scenelets can then be defined

as a function of the placement P and the scenelet assignment $\mathscr{X}$,

$$\widehat{q}_k^t(P,\mathscr{X}) = \sum_l \sum_{i=1}^{\max(t,n_l)} \mathscr{X}_{l(1+t-i)}\, T(P_{(1+t-i)}) s_k^{li}, \tag{6.6}$$

where s_k^{li} is the 3D position of joint k in frame i of scenelet l and $T(P_t)$ is the transformation due to placement P_t.

Finally, the objects in the scene are obtained from all scenelets that have been assigned to the scene as,

$$O(P,\mathscr{X}) = \bigcup_{\{(l,t)\ |\ \mathscr{X}_{lt}=1\}} T(P_t, O^l),$$

where we denote with $T(P,O)$ the transformation of objects in O to the placement P, *i.e.,* $T(P,O^l) = \{(T(p),\kappa,b)\ |\ (p,\kappa,b) \in O^l\}$.

6.6.2 Video Segments with Low Interaction-likelihood

Here, we fit static skeletons to each frame as the segment likely contains unoccluded human performance without object interactions. Since the number of degrees of freedom for human poses is smaller than for human-object interactions, the space of possible human poses can be covered more accurately than the space of possible human-object interactions. Thus, fitting static skeletons to the video gives better performance in unoccluded sequences that do not contain interactions.

The aforementioned 3D human pose reconstruction methods [160, 130] retrieve the best matching 3D skeleton pose for a given frame. Such pose, however, is defined in the *local* space of the skeleton and does not give us the placement of the skeleton in the scene. We fit the retrieved 3D skeleton to our video by optimizing the 3D placement of the skeleton, *i.e.,* the variables are only placement attributes of the local poses. We fit skeletons only to frames that do not have any scenelet assignment, *i.e.,* $\eta_t = 0$. In such frames, the joint locations $\breve{q}_k^t$ for video sequences are then defined as,

$$\breve{q}_k^t(P,\mathscr{X}) = (1-\eta_t)\, T(P_t) a_k^t, \tag{6.7}$$

where the first term is only non-zero if no scenelet is assigned to frame t, and a_k^t is the local skeleton pose computed by using Tomè et al. or LCR-Net++, $P_t = (x,y,z,\theta)$ is

the placement of the skeleton in frame t, and $T(P_t)$ is the transformation to placement P_t. Combining Equations 6.6 and 6.7, we define the location of any joint q_k^t in the video as,

$$q_k^t(P,\mathscr{X}) = \hat{q}_k^t(P,\mathscr{X}) + \check{q}_k^t(P,\mathscr{X}). \tag{6.8}$$

In the following, we will omit the explicit dependence of $q_k^t(P,\mathscr{X})$ and $o_i(P,\mathscr{X})$ on P and $\mathscr{X}$ for a less cluttered notation.

6.6.3 Scene Mapping via a Global Optimization

We have now set up our search space over possible configurations of objects and actor motions, parameterized through the scenelet and pose placements P and the assignment matrix $\mathscr{X}$. We define an energy in this space that can be minimized to obtain a consistent configuration of objects and actor motions given the observations in the video as,

$$L_{\text{global}}(\{u_k^t\},\mathscr{X},P) = L(\{u_k^t\},\mathscr{S},P) + w_s L_s + w_c L_c + w_m L_m, \tag{6.9}$$

where the first term denotes how well the current configuration explains the image-space joint detections as described in Equation 6.1. L_s encourages *smoothness* among human performance; L_c penalizes *intersections* between objects; and L_m penalizes *intersections* between the motion clip and objects.

Smoothness term (L_s). The smoothness term ensures continuity of the synthesized motion by measuring the finite difference approximation of time derivative of the synthesized joint locations as,

$$L_s = \sum_t \left\| q_\lambda^t - q_\lambda^{t-1} \right\|_2^2, \tag{6.10}$$

where λ is the index of the pelvis joint at video time of frame t.

Object intersection term (L_c). The object intersection term discourages object-object penetration. In our flat scene assumption, all objects are placed on the ground plane. We approximate intersections in 2D, using the projections of objects to the ground plane. Again, to obtain non-zero gradients, which are necessary to resolve intersections in a gradient-based solver, we quantify the amount of penetration using signed distance



Figure 6.5: Interaction-guided scene layout Consistent object layout and human movement as predicted by iMAPPER on various monocular videos. (Bottom-row) For qualitative evaluation, we overlay, shown from top-view, estimated scene layout versus annotated ground truth. For quantitative evaluation, please refer to Table 6.1 and 6.2. Please refer to the supplementary material of the project for videos and results.

functions as,

$$L_c = - \sum_{b_i \neq b_j \wedge \theta_i \neq \theta_j} \left(\int_{\Lambda(o_i)} \delta^-_{o_j}(x)\, dx + \int_{\Lambda(o_j)} \delta^-_{o_i}(x)\, dx \right) , \tag{6.11}$$

where $\delta^-_{o_i}$ is the negative part of the signed distance function of object o_i, $\Lambda(o_i)$ is the projection of object o_i to the ground plane, x is a point on the ground plane, b_i is the label of object o_i, and θ_i its orientation. We do not penalize intersecting objects that have the same label and orientation, since we assume these to be representations of the same object placed by different scenelets.

Motion intersection term (L_m). The motion intersection term discourages humans going through objects. Specifically, the trajectory of the human motion provides information about empty regions in the scene. For efficiency, we compute the intersection in 2D on the ground plane and focus on three joints only: the pelvis joint and the two knee joints. In practice, we have found that taking the maximum 2D distance of these three joints to objects allows a reasonable estimation of full 3D intersections in our scenes. We use,

$$L_m = \sum_t \max_{q \in \{q^t_\lambda,\, q^t_{\Gamma l},\, q^t_{\Gamma r}\}} \min_i \delta_{o_i}(q), \tag{6.12}$$

where δ_o is the signed distance function of object o, and $q^t_{\Gamma l}$, $q^t_{\Gamma r}$ are the left- and right-knee joints, respectively.

Global optimization.. We obtain the final solution as,

$$\mathscr{X}^{\star}, P^{\star} = \underset{\mathscr{X},P}{\operatorname{argmin}} L_{\text{global}}(\{u_k^t\}, \mathscr{X}, P). \tag{6.13}$$

We can then utilize the optimized assignment $\mathscr{X}^{\star}$ and placement $P^{\star}$ to extract joint locations q_k^t and the placements of objects o_i from selected scenelets and skeletons.

The above optimization is challenging given the mix of discrete and continuous parameters, and a highly non-linear energy function. We simplify the task by progressively performing the optimization.

(i) First, we estimate *interaction-saliency* for all frames of the video (as described in Section 6.4.2).

(ii) Next, starting with a smaller set of candidates ($\mathbb{K}_{\text{loc}}$) for the high-interaction likelihood video segments (as described in Section 6.5), we evaluate a selection of computationally-efficient energy terms (Equations 6.2, 6.3, 6.10, 6.12) locally.

(iii) Finally, by committing to a smaller set of high-scoring candidates ($\mathbb{K}_{\text{glob}}$, as described in Section 6.6), we optimize placements P of all fitted models in the scene, both local skeletons and scenelets, using the full energy term. Starting from the local optima from (ii), we optimize Equation (6.13), which translates to adding Equation 6.11 and inter-scenelet versions of Equations 6.3 and 6.12 to the energy function.

The scenelet assignment $\mathscr{X}$ is optimized indirectly by filtering out candidates in each stage of the decomposed optimization instead of immediately invoking an integer program. For simpler scenes, we perform one optimization for all combinations of the few remaining candidates and keep the combination that results in the scene with the lowest fitting energy.

For each of the three stages, described respectively in Sections 6.4.2, 6.5 and 6.6, we perform an optimization (using the quasi-Newton solver L-BFGS-B [15]) over the placement parameters P. We implemented the optimization in Tensorflow[3] and optimize using a Titan X (Pascal) with 12GB memory. The gradient tolerance termination criterion is set to its minimum value 10^{-12}. We optimize fitting all scenelets (≈ 1500) to a single frame at once with the terms in Equations 6.2 and 6.10 to estimate the interaction-likelihood score. For each local maximum of the interaction-likelihood

[3] `http://www.tensorflow.org/api_docs/python/tf/contrib/opt/ScipyOptimizerInterface`

function over time, we re-optimize the top $\mathbb{K}_{\text{loc}} = 200$ fits as described in (i) in batches of 10-25 due to the larger memory requirements of our implementation of Equation 6.3.

6.6.3.1 Keypoint Confidence using LCR-Net++

When using LCR-Net++, we estimate confidence of the 2D detection keypoints as

$$v_k = \underset{q_k^i \in pose\ proposals}{var} \left(q_k^i\right) / \left(1 + \exp\left(-0.2s' + 3.5\right)\right)$$

$$c_k(v_k) = 1/\left[1 + \exp\left\{-10\exp\left(\log(v_k)/P_{99}\left(\log(v_k)\right)\right) + 20\right\}\right]$$

where, var_k denotes the variance of the 3D joint position among the grouped pose proposals, and P_{99} denotes the 99th percentile of *log* joint variances over the whole recording, assigning high confidence to low variance joint estimates, and s' is a per-pose score defined in Equation 6 in [130].

6.6.4 Multi-person scenes

In case of multi-person videos, I require the user to inspect the detected 2D skeletons over time and mark the ones that need tracking in the first frame they appear. The input pose detections are not temporally consistent, so I perform naïve actor-reidentification in the form of an MRF optimization. Specifically, given the number of actors n_a, the set of skeletons $S(t)$ detected in frame t of the video, the goal is to find the values of the binary variables $\xi^t_{a\to s} \in \{0,1\}$, where $\xi^t_{a\to s} = 1$ if actor a is associated with skeleton $s \in S(t)$. To allow some actors to be identified as *invisible* in certain portions of the video, I define a dummy skeleton $\bar{s}$ such that if $\xi^t_{a\to\bar{s}} = 1$, a is marked as invisible. I represent the user annotations for a new actor in the first frame they appear as hard constraints in the optimization. In addition, I define unary and binary terms. The unary term, $E_u(a,s,t)$, measures the cost of assigning an actor a to a skeleton s at a frame t and is based on per-joint confidence measures, $c^t_{k,s}$:

$$E_u(a,s,t) = -\xi^t_{a\to s} \begin{cases} \frac{1}{n_{joints}} \sum_k^{n_{joints}} c^t_{k,s} & if\ s \neq \bar{s} \\ 1 & else. \end{cases} \tag{6.14}$$

Assigning the dummy skeleton, $\bar{s}$, to any actor results in a fixed cost of 1. The binary term, $E_b(a,s_0,s_1,t)$, measures the cost of assigning an actor a to the skeleton s_0

and s_1 in frames t and $t+1$ respectively:

$$E_b(a,s_0,s_1,t) = \xi^t_{a\to s_0}\xi^{t+1}_{a\to s_1}C(a,s_0,s_1,t) \tag{6.15}$$

$$C(a,s_0,s_1,t) = \begin{cases} 1 & \text{if } \bar{s} \in \{s_0,s_1\} \\ \frac{1}{n_{joints}}\sum_k^{n_{joints}} \left\| \frac{u^t_{k,s_0} - u^{t+1}_{k,s_1}}{diag} \right\|_2^2 c^t_{k,s_0}c^{t+1}_{k,s_1} & \text{else,} \end{cases}$$

where *diag* refers the the half of the diagonal of the image and transitioning from/to the dummy skeleton results in a fixed cost of 1. Finally, I optimize the following energy function:

$$\begin{aligned} \underset{\xi}{\text{argmin}}\, E = &-\sum_t \sum_{a\le n_a, s\in S(t)} E_u(a,s,t) \\ &+ w_{pw}\sum_t \sum_{a\le n_a} \sum_{s_0\in S(t)} \sum_{s_1\in S(t+1)} E_b(a,s_0,s_1,t) \end{aligned} \tag{6.16}$$

subject to the constraints:

$$\forall_t \forall_s \sum_{a\le n_a} \xi^t_{a\to s} = 1\,, \quad \forall_t \forall_a \sum_{s\neq\bar{s}} \xi^t_{a\to s} \le 1. \tag{6.17}$$

I set the relative weight of the binary term in Equation (6.16) as $w_{pw} = 10^3$ and optimize using a binary discrete optimizer (Gurobi).

For scenes with lots of occlusion and crossings, we require additional manual constraints. Once, multi-actor/2D skeleton associations are established, we optimize for 3D poses by enforcing smoothness between poses that belong to the same actor only.

6.7 Results and Discussion

We tested iMAPPER on a range of input monocular videos of varying complexity. For each of these benchmark videos (recorded in-house), we also manually annotated ground truth object placements, 3D actor poses, and action labels. Table 6.1 shows statistics of these videos while Figure 6.5 shows some examples. After we qualitatively discuss some results, we report how ground truth was annotated and explain our evaluation protocol.

Table 6.1: Performance statistics. List of scenes presented showing frame count (ct., N), fraction of frames with occlusion (frac., η), number of objects in the scene (obj., s_o) and number of objects with interactions (s_i). For comparison, we list number of objects with interactions detected by Mask-RCNN (MR, n_{MR}) and by iMAPPER (iM, n_{iM}). Also, we show quality of depth estimation error in cm by ground truth mask + MonoDepth (GT+MD, $\mu_{\text{GM}}(\sigma_{\text{GM}})$) and iMAPPER as mean (s.d.) (iM, $\mu_{\text{iM}}(\sigma_{\text{iM}})$) compared against ground truth annotations. Note that the scenes are ordered based on increasing difficulty which we assess by higher $N * \eta$ value.

scene	ct. N	frac. η	obj. $s_i(s_o)$	MR n_{MR}	iM n_{iM}	GT+MD $\mu_{\text{GM}}(\sigma_{\text{GM}})$	iM $\mu_{\text{iM}}(\sigma_{\text{iM}})$
Scene1	80	0.00	1 (2)	2	1	120 (-)	**45** (–)
Scene2	49	0.84	2 (4)	2	2	197 (24)	**135** (13)
Scene3	130	0.57	2 (4)	4	2	120 (10)	**72** (06)
Scene4	115	0.77	2 (2)	2	2	203 (12)	**70** (57)
Scene5	120	0.82	1 (1)	1	1	191 (-)	**25** (–)
Scene6	148	0.93	3 (4)	2	3	230 (52)	**72** (41)
Scene7	254	0.65	3 (3)	1	3	243 (98)	**70** (55)
Scene8	182	1.00	4 (4)	2	4	229 (48)	**53** (49)
Scene9	189	0.98	4 (4)	2	4	216 (50)	**69** (38)
Scene10	224	0.97	4 (4)	1	4	259 (101)	**51** (21)
Scene11	539	0.49	6 (11)	5	6	174 (73)	**83** (29)
Scene12	380	0.84	5 (5)	4	5	98 (36)	**57** (32)
Scene13	348	1.00	7 (14)	8	7	263 (117)	**81** (43)
Scene14	430	0.86	5 (5)	5	5	242 (68)	**58** (28)
Scene15	600	0.98	12 (14)	7	12	287 (120)	**68** (24)
μ	252.5	0.78	4.1 (5.4)	3.2	4.1	205 (62.2)	**67** (33.5)

Next, we provide quantitative evaluations, comparisons against baseline methods, and an ablation study. Please refer to the supplementary material of the project for the full input videos and annotations.

Qualitative evaluation

We show our results on some of the 15 different scenes from the Occluded Interaction Benchmark Dataset (see Section 6.7.1) in Figures 6.1 and 6.5. For each scene, we show a reference video frame. The recovered trajectory of the actor's pelvis is shown as a colored line and the colored skeleton shows an occluded actor pose. To help assess that iMAPPER finds consistent object arrangements and actor poses for occluded parts of the scene, we also show renderings of our results from the top view along with overlaid ground truth object locations. To better visualize the detected orientation of objects, we use object bounding boxes to scale and place category specific proxy object geometries in our scenelets – please note the meshes are *not* output by our method. Please refer

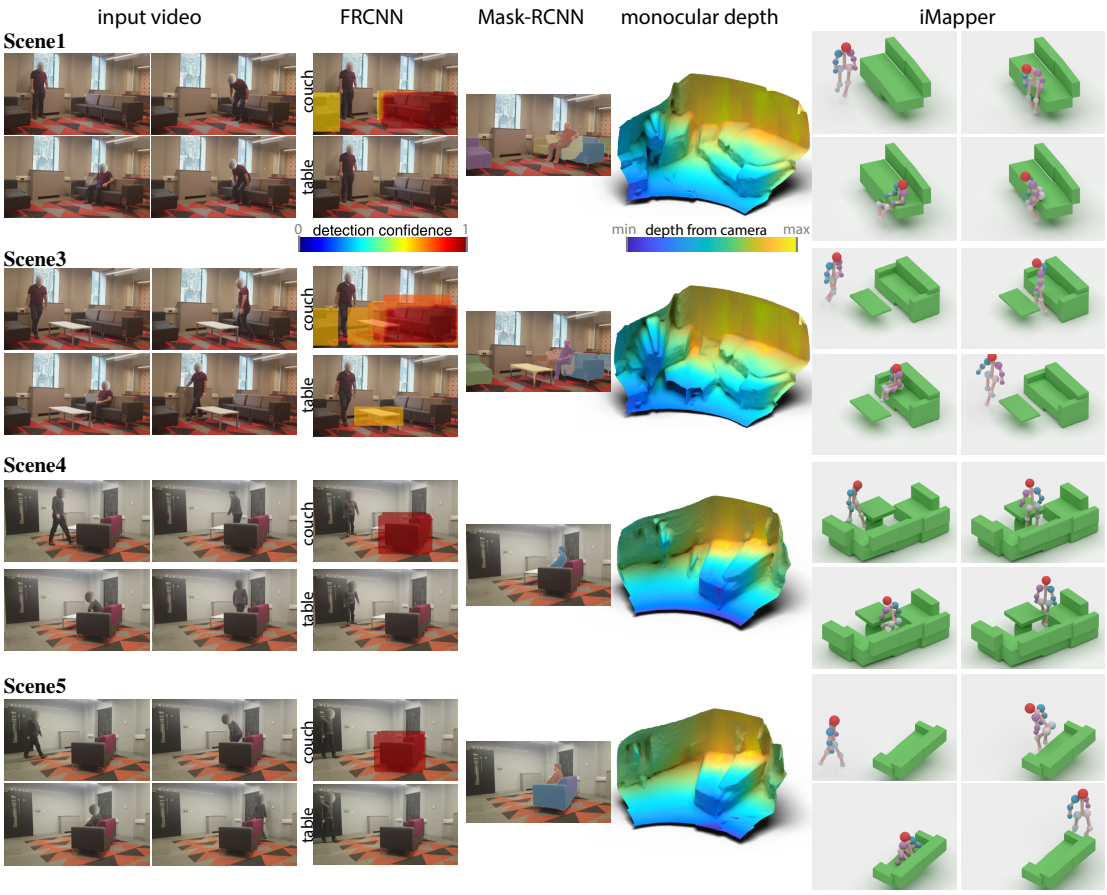


Figure 6.6: Object detection comparison. Qualitative comparisons to state-of-the-art object detection methods. Note that FRCNN and Mask-RCNN both produce only 2D image-space segments. iMAPPER synthesizes 3D scenes that plausibly match the video, even in sequences where the objects and the human performance are occluded. Encircled in red in the bottom depth map are the typical errors near depth discontinuities obtained from image depth estimators.

to supplementary material of the paper for full videos of the input and the generated scenes.

Note how much information is encoded in the interactions, enabling consistent reconstruction of the original scenes. Although we cannot hallucinate objects that are not interacted with, the quality of the generated scenes improves over time as more interactions 'reveal' the true underlying scene. As shown in Figure 6.12, as the same environment is explored over time, our system recovers larger parts of the object arrangement. Further, perturbations to the input (*e.g.,* in the form of the same action being performed by different people, or at different times) lead to slightly different, but still realistic and consistent scenes.

Multi-actor scenes. We described our method in the context of single actors. However, for scenes with multiple actors, our method generalizes easily when input

2D tracks come with correspondence information over time. We developed a semi-automatic labeling method that optimizes a Markov Random Field (MRF) energy term to find correspondences between skeleton detections, given some manual guidance (see Section 6.6.4 for details). This labeling tool was employed for the following multi-person scenes: Scene15, Library2, Angrymen, and Grease. Qualitative results can be seen in Figure 6.13.

6.7.1 Occluded Interaction Benchmark Dataset (i3DB)

We created a benchmark dataset consisting of 15 scenes by semi-automatically annotating both object locations and 3D world-space actor poses for each input video. To the best of our knowledge, this is the first benchmark dataset for scenes with medium to heavy occlusion containing both object and 3D human pose annotations over time.

6.7.1.1 Object location annotations

For *object locations*, we physically measured the dimensions of the scene objects and positioned objects as collections of oriented bounding boxes in the ground truth scene to minimize video reprojection error while using known camera intrinsics. We also added class labels (*e.g.,* `chair`, `table`, `shelf`) for each individual object. To evaluate the performance of iMAPPER and alternative methods in terms of object placement quality, we define two metrics (see Section 6.7.2 for comparison results).

Object Location Measure 1 (OBJCT). We define the metric OBJCT that counts the number of objects detected along with correctly labeled object class. We consider objects of interest as those that are participating in at least one interaction during the recorded video.

Object Location Measure 2 (OBJPOSN). We define the metric OBJPOSN that measures the mean and standard deviation of the distances between the predicted and the ground truth object centroids in the scene.

6.7.1.2 Human pose annotations

For human poses, we used an assisted approach to generate the ground truth since manually annotating 3D poses in each frame is not feasible for 1000s of frames. We started with estimated 2D joint locations [174] and then manually corrected them. These corrected 2D locations were then lifted first to local 3D space by running [160] – which works well in the absence of occlusion – and then to world space 3D using



Figure 6.7: Human pose detection comparison. Qualitative comparisons to Tomè et al. [160], which produces image-space and local 3D poses. Note that Tomè and colleagues do not compute world space positions of the skeletons. For better comparison, we position them in world coordinates using the hip locations as estimated by iMAPPER. Relevant differences between the methods are marked with red arrows. Note that our method gives realistic skeleton poses in many cases whereas the method of Tomè et al. fails, especially in occluded areas.

the reprojection and smoothness energy terms described in Section 6.5. Finally, we inspected the output 3D path, moved the skeletons to fit the ground truth scene layouts from Section 6.7.1.1 and added these corrections as additional constraints for the hip joints to the optimization. This process was repeated until we found no more significant errors. The number of corrections depended on the amount of occlusion in the scene, typically ranging from 10-50. When Tomè et al. [160] results deviated significantly from the actual human pose, we manually corrected the 3D poses using Blender. Finally, we used ray-tracing to establish ground truth visibility of the 3D joints using the oriented bounding boxes of the ground truth scene layouts.

To evaluate the performance of iMAPPER and alternative methods in terms of 3D human performance reconstruction quality, we define two metrics (see Section 6.7.3 for comparison results).

Human Pose Measure 1 (POSEEST). As a direct measure, we report the root mean square error over 14 joint locations that are common across the annotated 2D poses and 3D poses iMAPPER recovers. We compute this error both in local 3D (using origin at pelvis) space ('LC') and world space ('WC'), where available.

Human Pose Measure 2 (ACTNDETN). As an indirect measure, we train a pose-based action recognition network and evaluate how well the output poses can predict the ground truth action labels manually annotated at each frame of the input video. Specifically, we use the pose-based action recognition network proposed by Luvizon et al. [97], which takes as input 3D human poses (represented as 16 joints) for each video clip (we use a window size of 20 frames) and outputs the probability of each action class. The original network is trained using the NTU dataset [144] that provides 60 action classes. Most of these action classes are performed both in sitting and standing positions (*e.g.,* eating a snack, brushing teeth) and are not covered by the scenelet dataset we use [139]. In order to avoid any ambiguities, we retrain this network using action classes that are included both in our and the NTU dataset (*i.e.,* `walk`, `sitDown`, and `standUpFromSittingPosition`).

6.7.2 Evaluating Object Placement Quality

Object Location Measure 1 (OBJCT). We compare our method to per-frame region detection methods, FRCNN [129] and Mask-RCNN [60]. We show qualitative comparisons in Figure 6.6, and quantitative results in terms of metric OBJCT are given for Mask-RCNN in Table 6.1 (column 'MR'). For Mask-RCNN, we count the number of objects where at least 50% of the object's region was detected and correctly labeled on average, over all the frames. Since FRCNN and Mask-RCNN are designed to detect visible objects, they naturally fail to detect any objects 'hidden' behind visible objects leading to a low number of detections. Other systems that rely on these methods as their primary building blocks will have similar problems in occluded regions. In comparison, iMAPPER detects all the objects that participate in an interaction even if they are occluded.

Object Location Measure 2 (OBJPOSN). We compare against per-pixel monocular depth estimation [17] using the second metric, OBJPOSN. Table 6.1 shows the mean and standard deviation of the distances between the predicted and the ground truth object centroids in the scene for monocular depth estimation (column 'GT+MD') and iMAPPER (column 'iM'). For the monocular depth map, we approximate the object centroid as the mean world position of all samples that are inside the manually annotated ground truth 2D region of an object (*i.e.,* providing an upper bound on region detectors). Objects without a single visible pixel are ignored. The depth map contains only limited information about partially or fully occluded objects, resulting in large errors. As shown in Figure 6.6, fourth column, even for visible regions the estimated depths are smoothed out and fail to capture the object specific layout. In contrast, iMAPPER produces consistent objects along with their spatial locations.

6.7.3 Evaluating Actor Pose Quality

Human Pose Measure 1 (POSEEST). Most monocular 3D pose detection methods compute only *local* 3D poses, *i.e.,* joint locations relative to pelvis, limiting our choice of baselines for our first metric. We compare to Tomè et al. [160], and LCRNet++ [130] which both output local 3D joint locations, but do not provide world-space coordinates. Therefore, we use our method to optimize and lift their local predictions to world space (referred to as Tome3D and LCRNet++3D, respectively) and compare them in Table 6.2. While errors in local coordinates are reported in cm units, we report the errors in world coordinates as a fraction of the top-view 2D diagonal of the axis-aligned box of the ground truth path to be invariant to scene sizes. We define five categories based on the amount of error: $\texttt{BAD}(> 25cm)$, $\texttt{FAIR}(20-25cm)$, $\texttt{FINE}(15-20cm)$, $\texttt{GOOD}(10-15cm)$, $\texttt{EXCELLENT}(< 10cm)$.

Both Tome3D and LCRNet++3D are designed to capture the actual poses seen in the input sequence and thus achieve reasonable performance on average. However, they often produce unrealistic poses in occlusion regions as seen in the supplementary video of the project. On the other hand, the goal of our method is to generate realistic poses even in highly occluded interaction regions. While showing the same interactions, the poses in the matched scenelets may differ from the ground truth poses resulting in lower accuracy (we distinguish between accuracy and plausibility). Nevertheless, the

Table 6.2: POSEEST evaluation. Comparison of iMAPPER against Tome3D and LCR-Net++3D. Note that original methods return only local coordinates (LC) and we use our optimization to lift them to world coordinates (WC). LC units are in cm, WC units are in fraction of the top-view 2D diagonal of the axis-aligned bounding box of the ground truth path.

	Tome3D		LCR-Net++3D		iMapper	
	LC	WC	LC	WC	LC	WC
Scene4	FINE 19.9 (+70%)	FAIR .197 (+0.2%)	FAIR 21.5 (+83%)	FAIR .196	**GOOD** 11.7	BAD .225 (+15%)
Scene5	FINE 18.5 (+13%)	FINE .103 (+11%)	FAIR 21.9 (+34%)	**GOOD** .093	FINE 16.4	FINE .123 (+32%)
Scene7	BAD 26.3 (+39%)	FAIR .166 (+108%)	FAIR 20.1 (+6%)	FINE .123 (+54%)	FINE 19.0	**GOOD** .079
Scene10	FINE 16.6	**GOOD** .083 (+30%)	FAIR 22.3 (+34%)	FINE .135 (+111%)	FINE 19.1 (+15%)	**GOOD** .064
Scene11	**GOOD** 12.5	FINE .139 (+32%)	**GOOD** 14.0 (+12%)	FINE .105	FINE 17.3 (+39%)	FINE .127 (+20%)
Scene12	**GOOD** 13.2	FINE .114 (+64%)	**GOOD** 13.3 (+1%)	**GOOD** .070	FINE 16.2 (+23%)	**GOOD** .080 (+14%)
Scene13	FAIR 23.9	FINE .111 (+96%)	BAD 25.9 (+8%)	**GOOD** .057	BAD 28.1 (+17%)	**GOOD** .059 (+4%)
Scene14	**GOOD** 11.7	**GOOD** .089 (+45%)	**GOOD** 12.8 (+9%)	**GOOD** .065 (+6%)	**GOOD** 14.8 (+27%)	**GOOD** .062
μ	**17.11**	0.117	18.33	0.095	18.47	**0.087**

accuracy of the poses generated by iMAPPER are still in $\texttt{FINE} - \texttt{GOOD}$ range. Qualitative comparisons to Tome3D in Figure 6.7 verify these observations. In non-occluded frames, their poses are very close to ours (up to the smoothness term L_s). In occluded frames however, Tome3D returns unrealistic poses while iMAPPER continues to generate consistent poses in alignment with the discovered interaction. Figure 6.8 shows how such errors evolve over time for the three aforementioned methods as well as the number of occluded joints in the ground truth.

We also compare to Vnect [104] that provides world-space 3D pose estimates. A qualitative comparison is given in Figure 6.9. Since we do not have direct access to their source code, we compare to this method quantitatively using the same scene as in Figure 6.9 which is from the *MPI-INF-3DHP* dataset [103], where a relatively high-quality ground truth is available. While Vnect achieves an RMSE error of .399 and 28.9 in world-space and local-space respectively, iMAPPER achieves a lower error of .235 and 23.5.

Human Pose Measure 2 (ACTNDETN). In Table 6.3, we compare the results of a pose-based action detection network [97] on pose sequences obtained by Tome3D, LCRNet++3D, and iMAPPER. We report precision and accuracy for the duration of

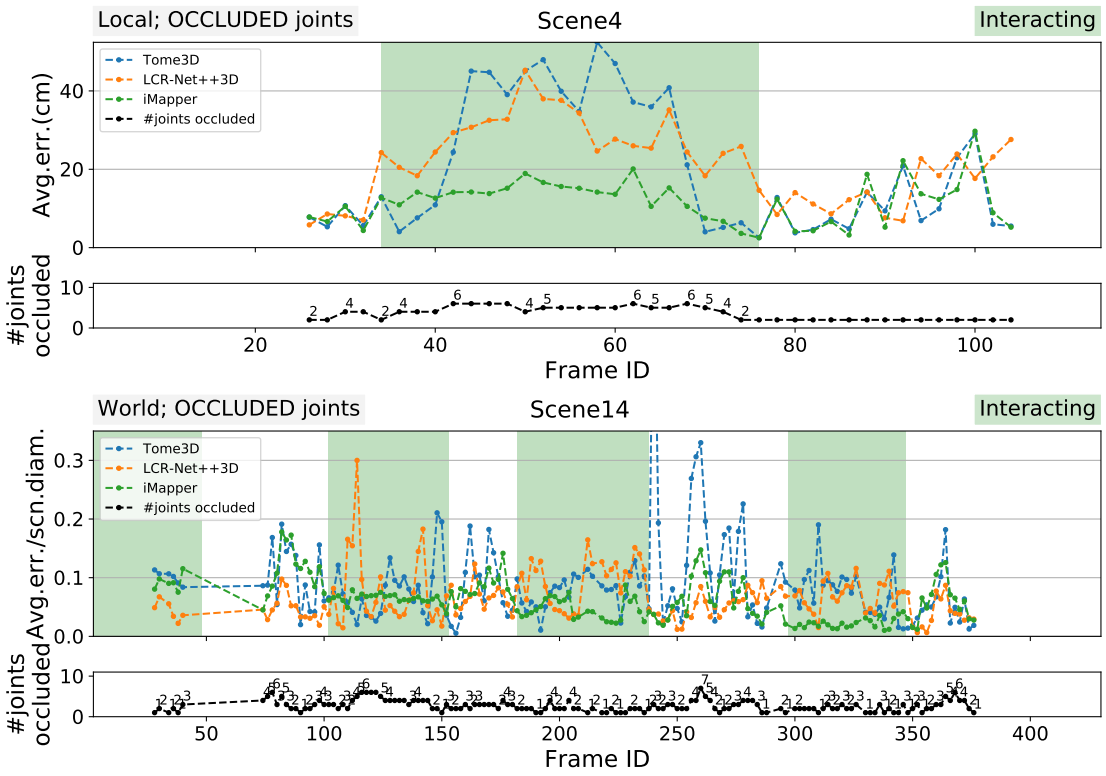


Figure 6.8: Comparison of the position accuracy of occluded joints over time. Green background denotes time intervals that contain interactions. Lower values are better. For world space, error metrics have been normalized by the top-view diagonal of the scene bounding box. Our algorithm detects and processes interactions and hence recovers from space-time segments with occlusions, as shown by the lower average error in green regions. However, outside interactions, the algorithm fits static skeletons to each frame, resulting in comparable performance to other methods. LCR-Net++3D tends to place ankle joints intersecting the floor during interactions, whilst Tome3D often yields physically not possible human poses. Note that although our error is not lowest in all frames, the poses detected by iMAPPER typically represent more consistent interactions, as shown in Table 6.3.

the sequence when interactions happen (by excluding the regions where the ground truth action label is walking) to focus evaluation around frames with occlusion. While Tome3D achieves reasonable precision, it does not produce reasonable 3D poses in occluded regions leading to low recall. While LCRNet++3D achieves higher recall, the estimated poses are often not accurate leading to low precision. In comparison, iMAPPER generally improves both in terms of precision and recall. Specifically, when we consider regions where interactions occur, iMAPPER improves by a large margin by recovering consistent poses from matching scenelets.

To verify that the scenes and human performance captures that iMAPPER generates are aligned with user expectations, we also conducted a comparative user study. For each question, we showed the users an input video of initial 2D pose estimates which are often

Table 6.3: ACTNDETN evaluation. Comparison of pose-based action detection [97] on human performance reconstruction by iMAPPER against those from Tome3D and LCR-Net++3D. We report both precision and recall for the parts of the input sequences that are annotated to be non-walking in the ground truth. We also report mean and (std. dev.) precision and recall across all the scenes. The corresponding pose-based action detection numbers for all frames (*i.e.,* walking or not) across scenes in this table are quite similar and included in supplementary material of the paper.

	Tome3D		LCR-Net++3D		iMAPPER	
	precision	recall	precision	recall	precision	recall
Scene4	0.00	0.00	0.00	0.00	**0.36**	**0.29**
Scene5	0.00	0.00	0.00	0.00	**0.13**	**0.09**
Scene7	0.07	0.01	0.13	0.12	**0.45**	**0.33**
Scene9	0.00	0.00	0.15	0.15	**0.57**	**0.57**
Scene10	0.08	0.03	0.11	0.11	**0.22**	**0.18**
Scene12	0.15	0.12	0.36	0.36	**0.64**	**0.51**
Scene13	0.00	0.00	0.14	0.14	**0.24**	**0.24**
Scene14	0.08	0.07	0.39	0.39	**0.50**	**0.50**
Overall	0.05 (0.06)	0.03 (0.04)	0.16 (0.15)	0.16 (0.15)	**0.39** (0.18)	**0.34** (0.17)

potentially noisy and miss joints in occlusion regions. Note that we do not show them the RGB videos as this is essentially also the input to iMAPPER and baselines. As output, we showed two different explanations of the input (*i.e.,* 3D scene and human performances) generated by iMAPPER and one of the two baseline methods (see Figure 6.10). For iMAPPER, we present the raw output results. For the two baselines, we accompanied the 3D pose trajectories obtained by Tome3D or LCRNet++3D with objects detected by

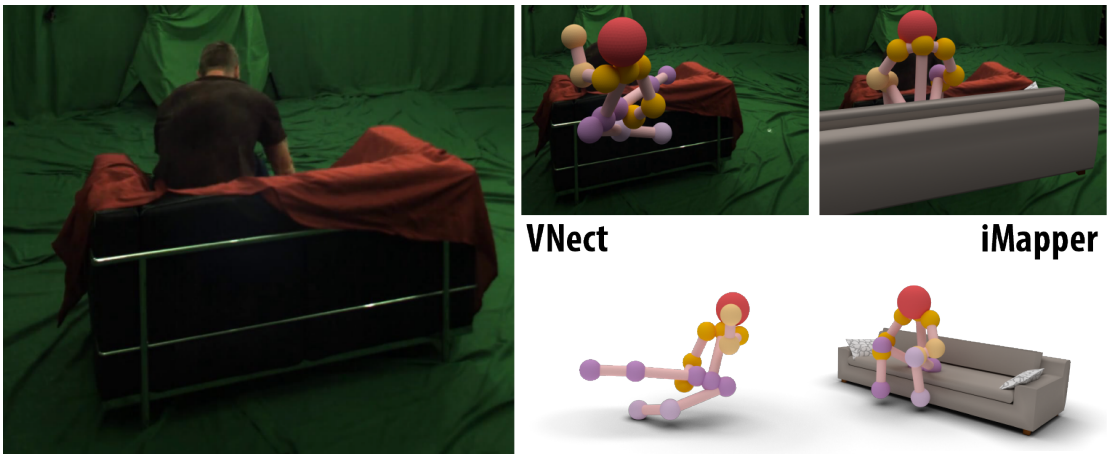


Figure 6.9: Comparison with VNect. State-of-the-art 3D human pose detection from monocular video VNect [104] breaks down in regions of occlusion (VNect was not designed to handle occlusions), while iMAPPER continues to produce realistic results because of explicit occlusion detection and handling. Bottom row shows the recovered human pose from another camera angle for better visibility.

MaskRCNN. For any object detected by MaskRCNN and matching the ground truth, we compute the 3D object centroid from a monocular depth estimate [17] and place the object to the centroid location by snapping it to the ground truth floor and manually orient the object based on the ground truth annotation.

6.7.4 User Study

We asked the users to select the method output that resembles the input the most judging based on how realistic the recovered interactions along with the involved objects are. We conducted the study for 5 scenes (Scene7, Scene9, Scene10, Scene13, Scene14) which resulted in a total of 15 pairs of queries (first and second ordering for any video was also flipped at random); 10 of these queries compared iMAPPER to alternatives whereas 5 queries compared the two alternative methods to each other. We had 25 unique users answer the queries for a total of 590 answers. We asked each query twice in random order to measure the consistency of the users and weighted the answers by the consistency. Users preferred iMAPPER output vs LCRNet++3D 99% the time and always preferred iMAPPER output over Tome3D. When comparing LCRNet++3D vs Tome3D, users preferred Tome3D 91% of the time. There is a strong user preference for iMAPPER outputs since the generated scenes are consistent with the input motion. In contrast, alternative methods most of the time result in missing objects involved in interactions due to occlusions, and lower quality of captured human performance, specifically in the occlusion regions.

6.7.5 Ablation Study

We report the effect of removing some of the terms from our optimization in Figure 6.11. For example, the occlusion term has a heavy influence if multiple joint locations are occluded in the video. Recall that our occlusion term is assymmetric, so that more occlusion always has less cost. The couch can thus be optimized to be closer to the camera to occlude more of the scene (the camera is at the bottom center of the image). The smoothness and reprojection terms also affect the result since they are used throughout the entire pipeline. Omitting the smoothness term can lead to path deformations, while leaving out the reprojection term removes the anchoring of the motions to the video and permits the smoothness term to possibly contract the path.

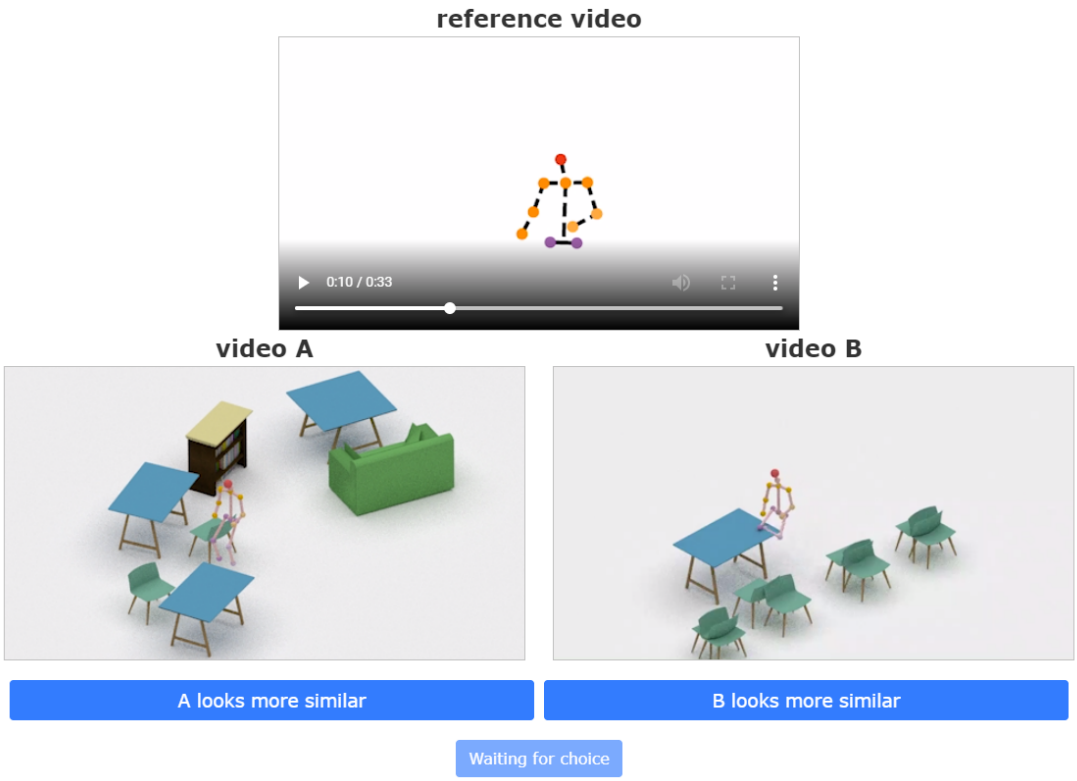


Figure 6.10: User study. We showed users an input video of initial 2D pose estimates and two different videos selected at random among the three explanations generated by iMAPPER and the two of the baseline methods.

Hold-one-out validation. We also evaluated iMAPPER on a hold out-set taken from the original PiGraphs dataset. We pick a single scene and remove the scenelets that were generated from this scene from our database, accounting for about 10% of our scenelets. We compare to the ground truth objects given in the PiGraphs dataset through manually established correspondence, and recorded deviations in world-space object placement using the OBJPOSN metric. The mean reconstruction error was 110 cm (s.d. 56 cm) with all relevant objects detected.

6.7.6 Limitations

We next discuss the current limitations of our method.

(i) In our formulation, we directly inherit human performance from the selected scenelets. Hence, we cannot correct any noise in the human motion as recorded in the interaction database. This can be addressed either by using a higher fidelity data capture or adaptively denoising the motion sequence.

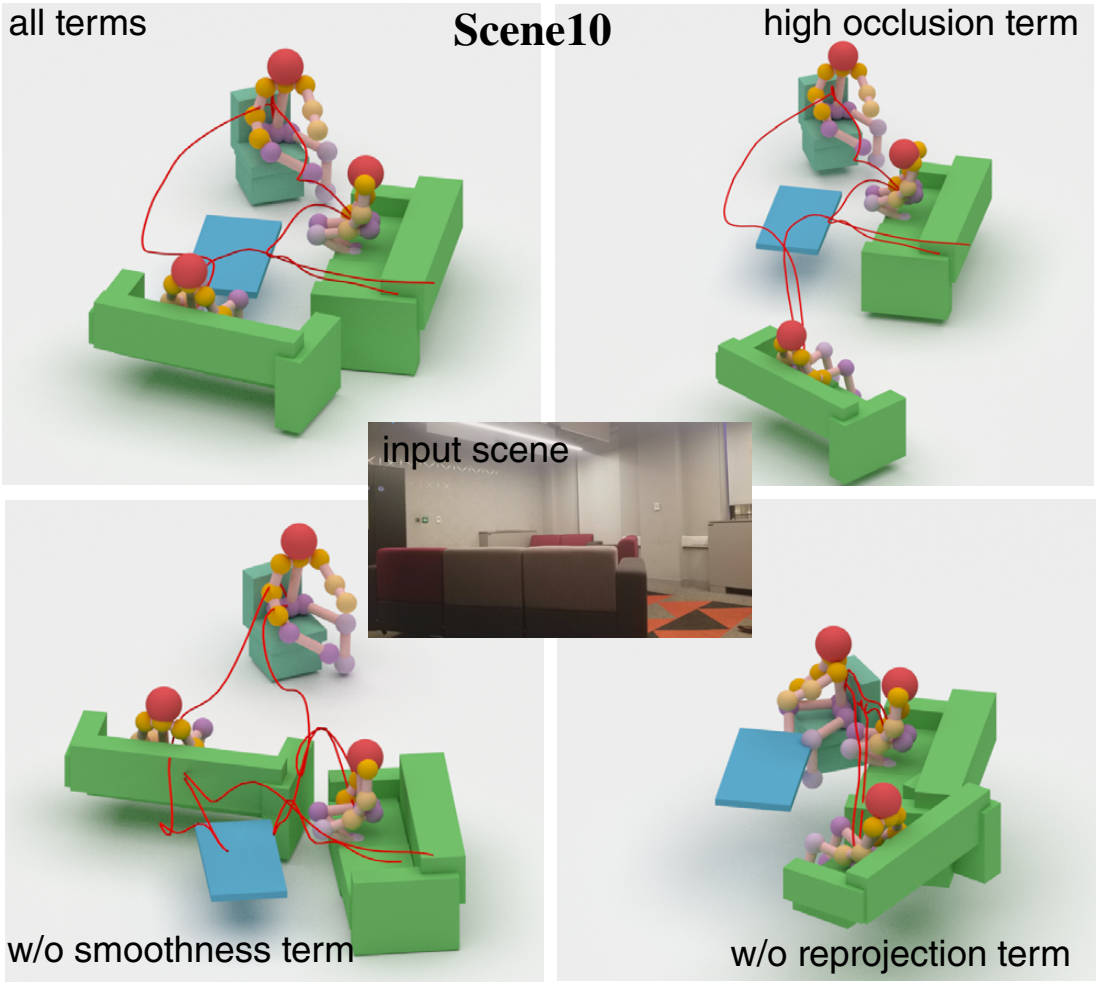


Figure 6.11: Ablation study. We test the effect of various terms on the solution for the scene shown in the center.

(ii) Since our method only uses 2D joint detections from the raw video and in absence of any image-space cues (*i.e.,* the objects are occluded), the method cannot provide any estimate about the size and extent of the objects, and hence cannot often distinguish between say a chair or a sofa.

(iii) Our method expects enough visible movement in the input video to trigger sufficient fraction of 2D joints to be detected. Otherwise, subsequent scenelet matching will fail, and in absence of matched scenelets, the object layout will remain incomplete. Similarly, occluded objects that remain un-interacted by the actor are likely to go undetected.

(iv) Finally, since our method builds on the expectation that people react similarly in similar settings, it will naturally get confused when this assumption is broken. For example, if a person decides to hand-walk, or use a sofa as a bed.

6.8 Conclusion and Future Work

In this chapter, I presented iMAPPER that takes as input a monocular footage of a person interacting with objects in a physical space and produces a realistic scene layout along with consistent human performances explaining the footage. iMAPPER detects and leverages human-object interactions in the input video to resolve medium to high level of occlusions that occur in such footage. At the heart of iMAPPER lies a novel data-driven method to assign *interaction-likelihood* scores to video segments that help identify space-time moments when matched human-object interactions provide reliable cues about the surrounding scene layout. The method retrieves corresponding interaction-salient scenelets as candidates fitted to the informative video segments, and then selects and positions them to form a global scene layout and 3D human pose estimates. I introduced the **i**3DB occluded interaction benchmark dataset for evaluating quality of interactions computed by different methods. The qualitative and quantitative evaluations demonstrate that iMAPPER produces realistic scene layouts as well as 3D pose estimates.

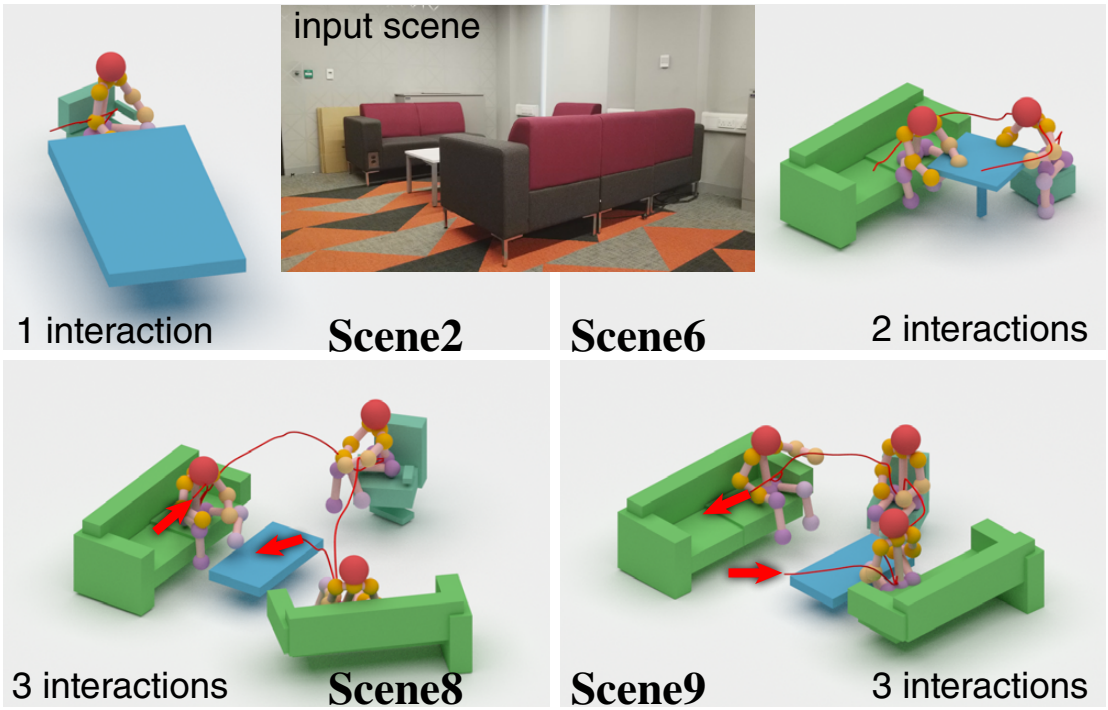


Figure 6.12: Progressive scene exploration. We show results of four different videos taken from the same scene. As interactions with more objects are made available, we can recompute the results to synthesize additional objects. Variations of scene explorations, for example performing the interactions in reverse order, as shown in the bottom row, give slightly different, but comparable and realistic results.

6.8.1 Future Directions

Exciting research directions lay ahead as we are only starting to capture, analyze, and understand the space of (human) interactions, or *interaction landscapes* (*c.f.,* [124]). Below I discuss some of the immediate issues.

Capturing richer interaction databases. Current datasets only capture limited variety of interactions, both in terms of different types of interactions and variance for each interaction type. For example, the method misses examples of interactions with small objects (*e.g.,* picking up a cup/glass, using pots and pans in kitchens, lifting a bag or suitcase), or examples of the different ways that people sit in sofas, couches and chairs. While significant progress has been made in capturing static environments at high geometric detail, capturing interactions remains *fundamentally* difficult because of heavy occlusion arising due to the interactions. One possibility is to separate the capture of static geometry (*e.g.,* with mobile 3D scanners) from the capture of interactions using a mix of sensors such as IMU sensors, RGBD scanners and markers.

Utilizing scene priors. So far the method used signals only from human-object interactions. However, in scenes with heavy occlusion, scenelet matching with partial (occluded) information may not be sufficient to accurately ground object positions. One possibility is to additionally use scene statistics and local context, as has been heavily utilized in scene synthesis research, to regularize the interaction-based layout reconstructions.

Recovering interactions over large timescales. As shown in Figure 6.12, iMAPPER has only a chance of recovering scene arrangements once people interact with parts of the environment. This suggests that the approach gets better as we 'observe' the scene over larger timescales, ideally days or weeks. However, then our static scene assumption breaks down as objects are going to be shifted and moved around. Hence, we would like to extend our approach to also capture space-time object movements, starting with rigid movement of objects, such as a moving chair.

Handling multiple actors. A next opportunity will be to improve iMAPPER to more accurately handle multiple actors. This will involve extending the 2D human pose detector to multiple actors by tracking instance correspondence over time using state-of-the-art person reidentification methods. Although the method naturally generalizes to this scenario, one has to obtain suitable priors to handle human-human interactions.

Figure 6.13 shows preliminary results. The method's limited capability of estimating scale arising from its monocular property is highlighted by the depth drift of sitting actors that mostly perform the same interaction during the whole recording. Adding further priors connecting interactions to realistic spatial motion statistics will further enhance the ouptut quality from such inputs.



Figure 6.13: Multi-actor scenes. The formulation of iMAPPER naturally generalizes to multiple actors allowing the retrieval of objects from scenes that don't exist anymore.

Chapter 7

Discussion

In the previous chapters, I presented a number of examples that confirm the main hypothesis of this thesis, *i.e.,* extending 3D scene analysis to auxiliary domains yields improvements in the quality of parameter estimation of the directly observable parts of the environment, and additional knowledge about the observed scenes including their not directly observable parts that previously were not possible to model as accurately. Naturally, there is lot of room for improvements and the conducted research opens up a wide selection of new avenues.

7.1 Regularity in raw scans

The main limitation of the method presented in Chapter 3 is its runtime due to the combinatorial optimization problem that has to be solved with a large number of variables. The main reason for the large number of variables in the current formulation is that the feature scale parameter ρ is a spatially invariant, fixed input to the method. In practice, for most types of 3D acquisition methods the sampling density of surfaces varies with the distance to the acquisition device and the relative angle between the surface normals and the view direction. As demonstrated, RAPter is excellent with varying sampling densities due to its non-local orientation formulation allowing multiple spatial locations to support the same orientation, however, keeping abundant samples in the problem translates to a lot of abundant computation. There are number of improvement opportunities with varying complexity.

The simplest to implement improvement is to introduce a split step for primitives similar to the merge step denoted 'simplification of co-planar primitives' in Algorithm 1, line 9. In such a split step the algorithm could evaluate, whether dividing a primitive's

points into two distinct groups each with their own new or existing orientation would result in decreasing or at least not increasing the energy of the solution more than by an ε parameter. Candidates to such a split step would be primitives with outlier data residual variance. The expected improvement in the output would be to prevent planes 'cutting off' corners due to the limitations of the region growing initialization. Another avenue of improvement would be making it more applicable to smooth pointclouds, *e.g.,* coming from structure-from-motion. These inputs are not well-suited right now to be processed by naïve region growing as due to the spatial smoothing of the preceding reconstruction algorithm, a well and evenly sampled curved surface can easily be "consumed" by a single primitive. Having a split step in the algorithm would allow the choice of the input spatial 'minimum feature size' parameter ρ not to be so critical and could be set to higher values. Primitives that in such cases initially would 'over-simplify' a part of the scene due to the simplicity of the region growing initialization could then be split up in parts of the scene where higher data-fidelity is necessary. An interesting extension in the same direction would be to assign a scale property to each primitive. Such a spatially varying scale parameter would allow the solution to adapt to the characteristics of the acquisition device. In the case of planes far away from any acquisition viewpoint the surface samplings tend to be much more noisy, especially when scanned with a Kinect®. In such cases, a larger scale parameter is needed for a single planar primitive to be able to accurately approximate the underlying surface. Such a large scale parameter is not suitable for parts of the scene that received more attention from the scanning viewpoints, and would reduce the accuracy capabilities of the algorithm. In practice, primitives with larger scale would need an additional energy term component that could encourage them to have larger spatial extent or smaller scale.

Combinatorial optimization is computationally expensive to solve. Naturally, reformulating to a simpler class of problems could yield great gains in efficiency. Equation (3.3) ensures, that orientation representative nodes are only actively contributing to the energy, if any 3D points are actually assigned to an orientation $\chi_{i\to j}$ that is represented by a representative χ_i. Replacing this quadratic term $(\chi_{i\to j}\chi_i - \chi_{i\to j})$ with a linear approximation would speed up the method significantly. Remember, the latter is a way to express the actual condition $\chi_i = \max_j \chi_{i\to j}$. Furthermore, recent advances in combinatorial optimization will also provide further improvements in runtime [48].

Data-driven methods represent a natural next step, when extrapolating improvement proposals to further horizons. Given sufficient amount of high-quality, annotated ground truth data, one can imagine the extraction of commonly occurring, scale invariant grammars of groups of relative arrangements of primitives, *e.g.,* as done in [94]. Since I have already made the assumption, that the algorithms in question are observing man-made scenes, it is reasonable to assume, that similar motifs of relative orientations occur probably across scales. *E.g.* top-view building outlines look like room outlines at larger scales. Learning these motifs of groups of primitives in before hand or even on the fly would allow the solution regularize noisy parts of the scenes using information learnt from well-sampled, less noisy parts of the scene.

7.2 Physical stability

There are three main limitations of the method, where I see room for improvement: i) manual initialization of floor, wall and ceiling, ii) varying density, and iii) per-group axis alignment of automatic initialization. Regarding i), the initialization of the static supports, plenty of methods exist for room estimation from RGB images [162], the extra depth information in the case of my RGBD input simplifies the question further [198]. Regarding ii) and iii), semantic segmentation [59] labeling the parts of the scene with object classes would allow the method to introduce priors on the density, dominant orientation of the objects and even typical relations (fixed, touching or none) between parts of the objects. Per-part density prediction might still be a hard task to solve, especially in the single-shot case where lots of parts would not be sampled well-enough to be recognizable by a semantic segmentation network trained on RGBD images. Perhaps a voxel classifying method could provide additional hints for initialization [163]. It would be interesting to see, how the performance of the above mentioned methods could further improve when augmented with knowledge about physical stability.

7.3 Accurate dynamics

Whilst it is remarkable how it is possible to ‘decode’ the information in the pre- and post-flight trajectories to not only reconstruct physically accurate collisions, but also to infer something about the internals of the objects, there are a few important limitations of the method, that will have to be addressed in the future. In the current formulation, an

important assumption was made, that the objects do not make any extra turns between two frames with orientation information. This means, that collisions with quick rotations are more difficult to accurately reconstruct. In the future, the method should be extended to take into account multiple possible hypotheses about the magnitude of the angular velocity.

An attempt was also made at removing the need for the 3D scans of the objects to be available as input to the method. I experimented with a structure-from-motion inspired formulation, that would solve for a fixed, sparse pointcloud for each object. The data to fit in the energy function originated from long-term 2D tracks of interest points on the object (see Brox and Malik [14]). It turns out, that with rotating objects, 2D tracks tend to strongly drift when the feature points disappear 'on the horizon', *i.e.,* when the surface they sit on gradually turns away from the camera view direction. Such feature drift adds too much noise to the problem, and this version of 'in flight reconstruction' was not successful. Perhaps shape silhouette based methods, such as [166] on a few, temporally distributed frames would provide enough information about the shape for the orientation regression to be successful.

Another important limitation of the method is that it represents the coefficient of restitution of an object as a single scalar variable. In practice, no objects are perfectly rigid, and most real-world objects have spatially varying material type and consistency. Such inhomogeneity in material results in high variance in the estimates of the coefficient of restitution, most likely correlated with where on the object the collisions happened. Modeling the restitution parameter with higher degrees of freedom will result in an explosion in the number of 'measurements' *i.e.,* input collision sequences necessary to be able to estimate the parameters of any such model. Deep networks are usually quite good at encoding implicit knowledge derived from large collections of data, and preliminary follow-up work (see Purushwalkam et al. [127]) has started to investigate the applicability of such methods to this problem.

As we learned in [31], compact mathematical representations of rotations are particularly difficult for deep networks to 'learn' to parameterize. An exciting future direction to solve the deep rotation parameterization problem is to, analogously to the coordinate lattice in Visual Interaction Networks [172], provide a form of orientation lattice to a network to augment the problem into an easier to parameterization space.

7.4 Human interactions

There are a few 'low-hanging fruits' in the selection of possible improvements to iMAPPER. A semantic- or instance-segmentation algorithm's output can be added to the input. The energy formulation should be updated to measure the intersection of the detected object masks and the candidate scenelets' objects' screen space reprojections. A discretized approximation similar to the one in Equation (6.11) fits the current implementation. Such an energy should be evaluated at the candidate selection step as well to filter out scenelet candidates that have category mismatch in their screen space reprojections.

Empty space is not explicitly modeled in the current formulation, and such modeling could further shrink the possible solution space efficiently. For a long-term scene build-up scenario, perhaps crowd walk-patterns could be analyzed following the work of Wang and O'Sullivan [170] in order to retrieve stronger evidence of empty space.

3D local poses are right now not optimized by iMAPPER. Tomè et al. [160] developed a mixture of PCA bases to represent the space of possible human poses with high fidelity. A natural extension to iMAPPER is to in an alternating optimization step use this mixture representation to slightly alter the 3D poses in the scenelets to optimize for more accurate reprojection compared to the input video. Some experiments using linear coefficients to add up the basis shapes of Tomè et al. were performed, but yielded worse than expected results, as the valid range of coefficients are not optimized for linear summation, but were designed to be parameterized by a neural network.

Deep scene occupancy prediction from evidence maps. In one attempt, I designed a deep learning based solution to predict the occupancy, object category and orientation of a top-view scene lattice based on input information derived from the input monocular video. In a preprocessing step, I followed the exhaustive fitting procedure described in Section 6.4.1 to retrieve the $\mathbb{K}_{\text{sal}}$ scenelets for each video frame. I could then use the fitting residual defined in Equation (6.5) to compute a weight for each scenelet match. The method would then 'splat' the top-view projected blobs of the different objects in the fitted scenelets to generate a 4D 'evidence map'. Each 2D slice of the map would represent an object category (simplified to 'sittable', 'walkable', 'table' in the first experiments) and an orientation (discretized to 16 slices). The grid cells of the 2D slice would store a scalar for how much 'evidence' of an interacting object was

accumulated during the scenelet matching phase. Object categories with ambiguous orientations (*e.g.*, tables) would contribute to multiple orientation slices. The resulting evidence maps marginalized over the orientation dimension for visualization can be seen in the first row of Figure 7.1. The input scene can be seen in Figure 6.12.

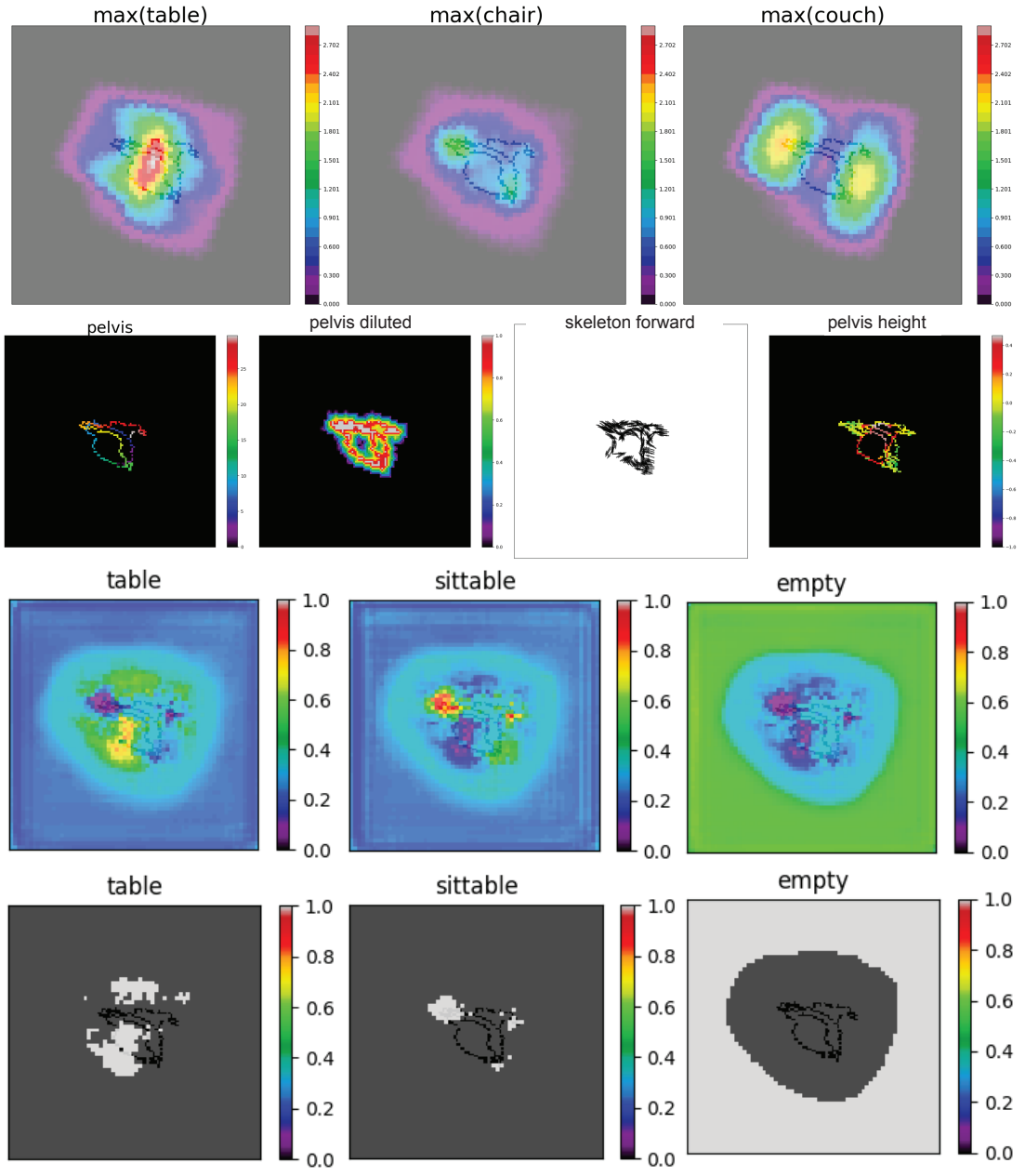


Figure 7.1: Deep scene occupancy prediction from evidence maps. (First row) accumulated evidence map marginalized over the orientation dimension for visualization. (Second row) Additional augmentations of the skeleton path and orientation. The colors in the left image ('pelvis') encode time in the input video. (Third row) Output occupancy predictions by the occupancy network. (Bottom row) Attempts at automatic thresholding of the predicted occupancy output.

I designed an hourglass network, that would take the above evidence volume of size $64 \times 64 \times (3 \cdot 16)$ and predict first, per-pixel occupancy in the top-view layout map and then, also an orientation for the same 2D lattice. Figure 7.2 shows the structure of the network.

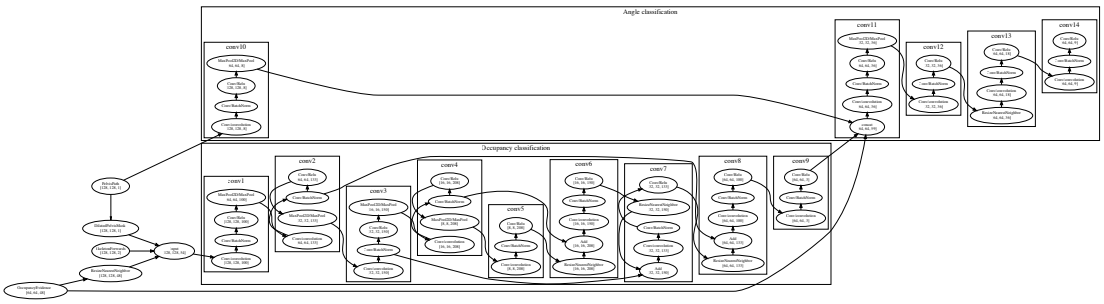


Figure 7.2: Deep scene occupancy prediction network. Per-category occupancy probabilities are predicted in the bottom branch. The intermediate occupancy output is then given to the orientation prediction network, that produces an additional output map of orientation probabilities.

The occupancy predictions were occasionally of promising quality, as shown in Figure 7.3. In general, only regions of the scene where the skeleton had actually visited had the chance to be labeled correctly. The orientation output maps were almost undistinguishable from random noise, and thus not included in the figures.

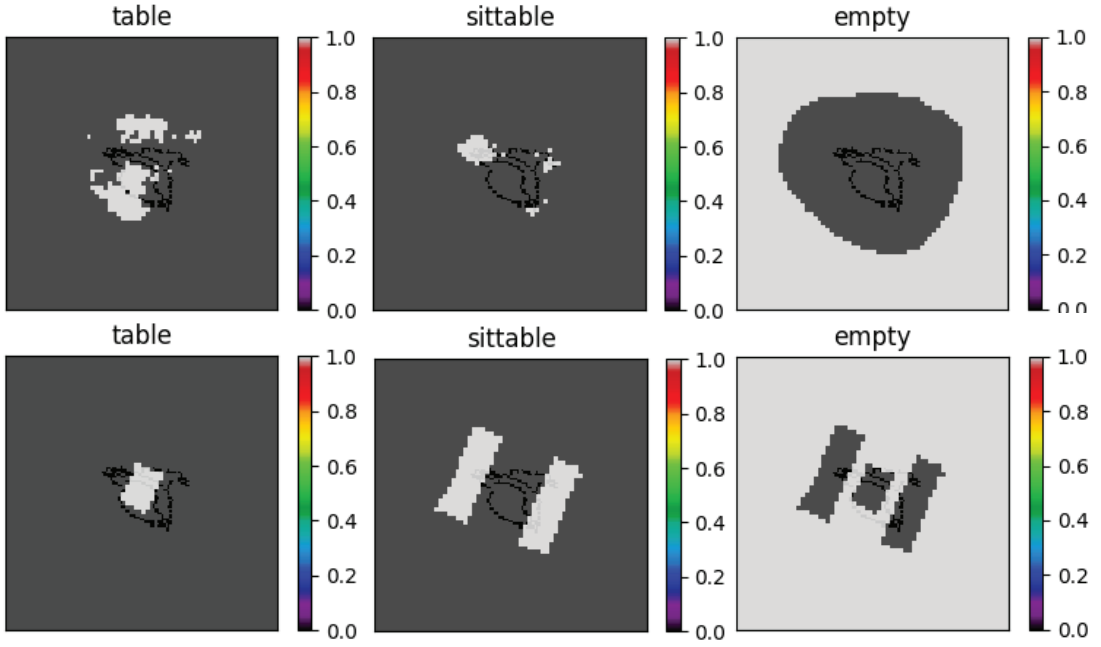


Figure 7.3: Deep scene occupancy prediction from evidence maps – comparison to ground truth. (top) Automatically thresholded per-category probabilities. (bottom) Ground truth per-category occupancy. It is encouraging to see, that sittable areas, where the skeleton had visited are correctly predicted to be sittable.

I concluded, that despite all augmentation efforts, the size of the dataset possible to build from the PiGraphs dataset was not sufficient for a network to be able to experience sufficient fraction of such a large solution space of object categories and orientations. It is understandable, that an orientation map of size $64 \times 64 \times 48$ has too many degrees of freedom compared to the dataset size (60 scene layouts split into training and test sets 90/10). It does not improve the conditions, that tables have ambiguous or symmetric 'forward' orientations.

7.5 Non-visual cues

In summary, I hypothesized, that when performing 3D scene analysis, it is advantageous to jointly reason about the not *directly* observable parts of the environment together with the observed regions yielding partial scans of the input scenes. I see the hypothesis confirmed: auxiliary domains of knowledge, such as perfect non-local regularity, stability, dynamic accuracy and consistent scene layout and human interactions can be added to the 3D scene analysis tasks in compact formulations. Furthermore, the parameterization of such models can be computed in real-world scenarios, and besides describing previously unaccessible parts of the scenes, the outcome of such computations tend to improve the estimates of the directly observable regions as well.

Bibliography

[1] AGARWAL, S., MIERLE, K., ET AL. Ceres Solver. http://ceres-solver.org/, 2016.

[2] AGUDO, A., MONTIEL, J. M. M., AGAPITO, L., AND CALVO, B. N. Online Dense Non-Rigid 3D Shape and Camera Motion Recovery. In *BMVC* (2014).

[3] ALAYRAC, J.-B., LAPTEV, I., SIVIC, J., AND LACOSTE-JULIEN, S. Joint discovery of object states and manipulation actions. In *IEEE ICCV* (2017).

[4] ANAND, A., KOPPULA, H. S., JOACHIMS, T., AND SAXENA, A. Contextually Guided Semantic Labeling and Search for Three-dimensional Point Clouds. *IJRR 32* (2013).

[5] ARIKAN, M., SCHWÄRZLER, M., FLÖRY, S., WIMMER, M., AND MAIERHOFER, S. O-Snap: Optimization-Based Snapping for Modeling Architecture. *ACM TOG 32* (2013).

[6] ARMSTRONG, W., AND GREEN, M. The Dynamics of Articulated Rigid Bodies for Purposes of Animation. *The Visual Computer 1* (1985).

[7] ATAER-CANSIZOGLU, E., TAGUCHI, Y., RAMALINGAM, S., AND GARAAS, T. Tracking an RGB-D camera using points and planes. In *Proceedings of the 2013 IEEE International Conference on Computer Vision Workshops* (2013).

[8] BAO, Y., CHANDRAKER, M., LIN, Y., AND SAVARESE, S. Dense Object Reconstruction Using Semantic Priors. In *IEEE CVPR* (2013).

[9] BARAFF, D. Curved Surfaces and Coherence for Non-penetrating Rigid Body Simulation. *ACM SIGGRAPH 24* (1990).

[10] BLUM, M., GRIFFITH, A., AND NEUMANN, B. A stability test for configurations of blocks. TR-AI Memo, 1970.

[11] BONAMI, P., ET AL. An Algorithmic Framework for Convex Mixed Integer Nonlinear Programs. *Discret. Optim. 5* (2008).

[12] BONGARD, J., AND LIPSON, H. Automated Reverse Engineering of Nonlinear Dynamical Systems. *Proceedings of the National Academy of Sciences 104* (2007).

[13] BOULCH, A., DE LA GORCE, M., AND MARLET, R. Piecewise-planar 3D reconstruction with edge and corner regularization. *CGF 33* (2014).

[14] BROX, T., AND MALIK, J. Object segmentation by long term analysis of point trajectories. In *ECCV* (2010).

[15] BYRD, R. H., LU, P., NOCEDAL, J., AND ZHU, C. A Limited Memory Algorithm for Bound Constrained Optimization. In *SISC* (1995).

[16] CAO, Z., SIMON, T., WEI, S.-E., AND SHEIKH, Y. Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields. In *IEEE CVPR* (2017).

[17] CHAKRABARTI, A., SHAO, J., AND SHAKHNAROVICH, G. Depth from a Single Image by Harmonizing Overcomplete Local Network Predictions. In *NIPS* (2016).

[18] CHANG, A., DAI, A., FUNKHOUSER, T., HALBER, M., NIESSNER, M., SAVVA, M., SONG, S., ZENG, A., AND ZHANG, Y. Matterport3D: Learning from RGB-D data in indoor environments. In *3DV* (2017).

[19] CHEN, K., LAI, Y.-K., WU, Y.-X., MARTIN, R., AND HU, S.-M. Automatic semantic modeling of indoor scenes from low-quality rgb-d data using contextual information. In *ACM SIGGRAPH Asia* (2014).

[20] CHENNEY, S., AND FORSYTH, D. A. Sampling Plausible Solutions to Multi-body Constraint Problems. *ACM SIGGRAPH* (2000).

[21] CHUM, O., AND MATAS, J. Matching with PROSAC: Progressive Sample Consensus. In *IEEE ICCV* (2005).

[22] CONOTTER, V., O'BRIEN, J. F., AND FARID, H. Exposing Digital Forgeries in Ballistic Motion. *IEEE Transactions on Information Forensics and Security 7* (2012).

[23] DAI, A., CHANG, A. X., SAVVA, M., HALBER, M., FUNKHOUSER, T., AND NIESSNER, M. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *IEEE CVPR* (2017).

[24] DAI, A., NIESSNER, M., ZOLLÖFER, M., IZADI, S., AND THEOBALT, C. BundleFusion: Real-time Globally Consistent 3D Reconstruction using On-the-fly Surface Re-integration. *ACM TOG 36* (2017).

[25] DAVIS, J., MARSCHNER, S. R., GARR, M., AND LEVOY, M. Filling Holes in Complex Surfaces Using Volumetric Diffusion. In *Proc. 3DPVT* (2002).

[26] DEL PERO, L., BOWDISH, J., KERMGARD, B., HARTLEY, E., AND BARNARD, K. Understanding bayesian rooms using composite 3D object models. In *IEEE CVPR* (2013).

[27] DELAITRE, V., FOUHEY, M., LAPTEV, I., SIVIC, J., GUPTA, A., AND EFROS, A. A. Scene Semantics from Long-Term Observation of People. In *ECCV* (2012).

[28] DOU, M., TAYLOR, J., FUCHS, H., FITZGIBBON, A., AND IZADI, S. 3D scanning deformable objects with a single RGBD sensor. In *IEEE CVPR* (2015).

[29] EBERLY, D. H. *Game physics*. Taylor and Francis, 2010.

[30] EHRHARDT*, S., MONSZPART*, A., J. MITRA, N., AND VEDALDI, A. Unsupervised Intuitive Physics from Visual Observations. In *ACCV* (2018). *joint first authors.

[31] EHRHARDT, S., MONSZPART, A., J. MITRA, N., AND VEDALDI, A. Taking Visual Motion Prediction To New Heightfields. In *Computer Vision and Image Understanding* (2019).

[32] EHRHARDT, S., MONSZPART, A., NILOY J. MITRA, AND VEDALDI, A. Unsupervised Intuitive Physics from Past Experiences. *arXiv e-prints* (2019). https://arxiv.org/abs/1905.10793.

[33] FIDLER, S., DICKINSON, S., AND URTASUN, R. 3D Object Detection and Viewpoint Estimation with a Deformable 3D Cuboid Model. In *NIPS* (2012).

[34] FISHER, M., RITCHIE, D., SAVVA, M., FUNKHOUSER, T., AND HANRAHAN, P. Example-based synthesis of 3D object arrangements. In *ACM SIGGRAPH Asia* (2012).

[35] FISHER, M., SAVVA, M., AND HANRAHAN, P. Characterizing structural relationships in scenes using graph kernels. In *ACM SIGGRAPH* (2011).

[36] FISHER, M., SAVVA, M., LI, Y., HANRAHAN, P., AND NIESSNER, M. Activity-centric Scene Synthesis for Functional 3D Scene Modeling. *ACM SIGGRAPH Asia 34* (2015).

[37] FOUHEY, D. F., DELAITRE, V., GUPTA, A., EFROS, A. A., LAPTEV, I., AND SIVIC, J. People Watching: Human Actions as a Cue for Single-View Geometry. In *ECCV* (2012).

[38] FRANK, B., RUHNKE, M., TATARCHENKO, M., AND BURGARD, W. 3D-reconstruction of indoor environments from human activity. In *IEEE ICRA* (2015).

[39] FU, L., ZHANG, J., AND HUANG, K. Beyond Tree Structure Models: A New Occlusion Aware Graphical Model for Human Pose Estimation. In *IEEE ICCV* (2015).

[40] FU, Q., CHEN, X., SU, X., AND FU, H. Pose-Inspired Shape Synthesis and Functional Hybrid. In *IEEE TVCG* (2017).

[41] FU, Q., CHEN, X., WANG, X., WEN, S., ZHOU, B., AND FU, H. Adaptive Synthesis of Indoor Scenes via Activity-Associated Object Relation Graphs. *ACM SIGGRAPH Asia 36* (2017).

[42] FURUKAWA, Y., CURLESS, B., SEITZ, S. M., AND SZELISKI, R. Manhattan-world stereo. In *IEEE CVPR* (2009).

[43] GALLUP, D., FRAHM, J.-M., MORDOHAI, P., YANG, Q., AND POLLEFEYS, M. Real-time plane-sweeping stereo with multiple sweeping directions. In *IEEE CVPR* (2007).

[44] GARG, R., ROUSSOS, A., AND AGAPITO, L. Dense Variational Reconstruction of Non-rigid Surfaces from Monocular Video. In *IEEE CVPR* (2013).

[45] GIBSON, J. *The Ecological Approach To Visual Perception*. Pschology Press, 1986.

[46] GILARDI, G., AND SHARF, I. Literature Survey of Contact Dynamics Modelling. *Mechanism and Machine Theory 37* (2002).

[47] GKIOXARI, G., GIRSHICK, R., DOLLÁR, P., AND HE, K. Detecting and Recognizing Human-Object Interactions. In *IEEE CVPR* (2018).

[48] GOTO, H., TATSUMURA, K., AND DIXON, A. R. Combinatorial optimization by simulating adiabatic bifurcations in nonlinear Hamiltonian systems. *Science Advances 5* (2019).

[49] GREGSON, J., THUEREY, N., IHRKE, I., AND HEIDRICH, W. From Capture to Simulation - Connecting Forward and Inverse Problems in Fluids. *ACM SIGGRAPH 33* (2014).

[50] GRZESZCZUK, R., TERZOPOULOS, D., AND HINTON, G. Neuroanimator: Fast Neural Network Emulation and Control of Physics-based Models. *ACM SIGGRAPH 1* (1998).

[51] GSCHWANDTNER, M. *Support Framework for Obstacle Detection on Autonomous Trains*. PhD thesis, Univ. Salzburg, 2013.

[52] GUO, R., AND HOIEM, D. Support Surface Prediction in Indoor Scenes. In *IEEE ICCV* (2013).

[53] GUPTA, A., EFROS, A. A., AND HEBERT, M. Blocks World Revisited: Image Understanding Using Qualitative Geometry and Mechanics. In *ECCV* (2010).

[54] GUPTA, A., KEMBHAVI, A., AND DAVIS, L. S. Observing Human-Object Interactions: Using Spatial and Functional Compatibility for Recognition. In *IEEE PAMI* (2009).

[55] GUPTA, A., SATKIN, S., EFROS, A. A., AND HEBERT, M. From 3D Scene Geometry to Human Workspace. In *IEEE CVPR* (2011).

[56] HARARY, G., TAL, A., AND GRINSPUN, E. Context-based Coherent Surface Completion. *ACM TOG 33* (2014).

[57] HARRIS, C., AND STENNETT, C. RAPID-a video rate object tracker. In *BMVC* (19990).

[58] HARTLEY, E., KERMGARD, B., FRIED, D., BOWDISH, J., PERO, L. D., AND BARNARD, K. Bayesian Geometric Modeling of Indoor Scenes. In *IEEE CVPR* (2012).

[59] HAZIRBAS, C., MA, L., DOMOKOS, C., AND CREMERS, D. FuseNet: Incorporating Depth into Semantic Segmentation via Fusion-Based CNN Architecture. In *ACCV* (2016).

[60] HE, K., GKIOXARI, G., DOLLÁR, P., AND GIRSHICK, R. Mask R-CNN. In *IEEE ICCV* (2017).

[61] HEDAU, V. Recovering free space of indoor scenes from a single image. In *IEEE CVPR* (2012).

[62] HEDAU, V., HOIEM, D., AND FORSYTH, D. Thinking Inside the Box: Using Appearance Models and Context Based on Room Geometry. In *ECCV* (2010).

[63] HU, R., VAN KAICK, O., WU, B., HUANG, H., SHAMIR, A., AND ZHANG, H. Learning how objects function via co-analysis of interactions. In *ACM TOG* (2016).

[64] HU, R., ZHU, C., VAN KAICK, O., LIU, L., SHAMIR, A., AND ZHANG, H. Interaction context (ICON): Towards a geometric functionality descriptor. In *ACM TOG* (2015).

[65] HUANG, C.-H., BOYER, E., NAVAB, N., AND ILIC, S. Human Shape and Pose Tracking Using Keyframes. In *IEEE CVPR* (2014).

[66] HUANG, J.-B., AND YANG, M.-H. Estimating Human Pose from Occluded Images. In *ACCV* (2009).

[67] HUANG, S.-S., FU, H., AND HU, S.-M. Structure guided interior scene synthesis via graph matching. In *Graphical Models* (2016).

[68] HUBER, P. J. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics 35* (1964).

[69] HUETING, M., REDDY, P., YUMER, E., KIM, V. G., CARR, N., AND MITRA, N. J. SeeThrough: Finding objects in heavily occluded indoor scene images. In *3DV* (2018).

[70] INSAFUTDINOV, E., PISHCHULIN, L., ANDRES, B., ANDRILUKA, M., AND SCHIELE, B. DeeperCut: A Deeper, Stronger, and Faster Multi-Person Pose Estimation Model. In *ECCV* (2016).

[71] ISACK, H., AND BOYKOV, Y. Energy-Based Geometric Multi-model Fitting. *IJCV 97* (2012).

[72] IZADINIA, H., SHAN, Q., AND SEITZ, S. M. IM2CAD. In *IEEE CVPR* (2017).

[73] JIA, Z., GALLAGHER, A., SAXENA, A., AND CHEN, T. 3D-Based Reasoning with Blocks, Support, and Stability. In *IEEE CVPR* (2013).

[74] JIANG, H., AND XIAO, J. A Linear Approach to Matching Cuboids in RGBD Images. In *IEEE CVPR* (2013).

[75] JIANG, Y., KOPPULA, H. S., AND SAXENA, A. Modeling 3D Environments Through Hidden Human Context. In *IEEE PAMI* (2016).

[76] KANG, C., AND LEE, S.-H. Scene reconstruction and analysis from motion. In *Graphical Models* (2017).

[77] KILIAN, M., MONSZPART, A., AND MITRA, N. J. String Actuated Curved Folded Surfaces. *ACM TOG 36* (2017).

[78] KIM, B., KOHLI, P., AND SAVARESE, S. 3D Scene Understanding by Voxel-CRF. *IEEE ICCV* (2013).

[79] KIM, V. G., CHAUDHURI, S., GUIBAS, L., AND FUNKHOUSER, T. Shape2Pose: Human-centric shape analysis. In *ACM SIGGRAPH* (2014).

[80] KIM, Y. M., MITRA, N. J., HUANG, Q., AND GUIBAS, L. Guided Real-Time Scanning of Indoor Objects. *CGF 32* (2013).

[81] KIM, Y. M., MITRA, N. J., YAN, D. M., AND GUIBAS, L. Acquiring 3D indoor environments with variability and repetition. *ACM SIGGRAPH Asia 31* (2012).

[82] KIM, Y. M., MITRA, N. J., YAN, D.-M., AND GUIBAS, L. Acquiring 3D Indoor Environments with Variability and Repetition. *ACM SIGGRAPH Asia 31* (2012).

[83] KLEPPNER, D., AND KOLENKOW, R. *Introduction to Mechanics, 2nd Ed.* Cambridge University Press, 2013.

[84] KOPPULA, H. S., ANAND, A., JOACHIMS, T., AND SAXENA, A. Semantic labeling of 3D point clouds for indoor scenes. *NIPS* (2011).

[85] KRASNER, L. *Environmental Design and Human Behavior*. Elsevier, 2013.

[86] LADICKY, L., RUSSELL, C., KOHLI, P., AND TORR, P. H. Graph cut based inference with co-occurrence statistics. In *ECCV* (2010).

[87] LAFARGE, F., AND ALLIEZ, P. Surface reconstruction through point set structuring. In *CGF Eurographics* (2013).

[88] LEE, D. C., GUPTA, A., HEBERT, M., AND KANADE, T. Estimating Spatial Layout of Rooms using Volumetric Reasoning about Objects and Surfaces. *NIPS 24* (2010).

[89] LEORDEANU, M., AND HEBERT, M. A Spectral Technique for Correspondence Problems Using Pairwise Constraints. In *IEEE ICCV* (2005).

[90] LI, Y., WU, X., CHRYSANTHOU, Y., SHARF, A., COHEN-OR, D., AND MITRA, N. J. GlobFit: Consistently Fitting Primitives by Discovering Global Relations. *ACM TOG 30* (2011).

[91] LI, Y., ZHENG, Q., SHARF, A., COHEN-OR, D., CHEN, B., AND MITRA, N. J. 2D-3D Fusion for Layer Decomposition of Urban Facades. In *IEEE ICCV* (2011).

[92] LI, Z., SEDLAR, J., CARPENTIER, J., LAPTEV, I., MANSARD, N., AND SIVIC, J. Estimating 3d motion and forces of person-object interactions from monocular video. In *IEEE CVPR* (2019).

[93] LIN, H., GAO, J., ZHOU, Y., LU, G., YE, M., ZHANG, C., LIU, L., AND YANG, R. Semantic Decomposition and Reconstruction of Residential Scenes from LiDAR Data. In *ACM SIGGRAPH* (2013).

[94] LIU, T., CHAUDHURI, S., KIM, V., HUANG, Q., MITRA, N., AND FUNKHOUSER, T. Creating consistent scene graphs using a probabilistic grammar. In *ACM SIGGRAPH Asia* (2014).

[95] LIU, Z., GORTLER, S. J., AND COHEN, M. F. Hierarchical Spacetime Control. *ACM SIGGRAPH 1* (1994).

[96] LOPER, M. M., AND BLACK, M. J. OpenDR: An Approximate Differentiable Renderer. In *ECCV* (2014).

[97] LUVIZON, D. C., PICARD, D., AND TABIA, H. 2D/3D Pose Estimation and Action Recognition Using Multitask Deep Learning. In *IEEE CVPR* (2018).

[98] MA, R., LI, H., ZOU, C., LIAO, Z., TONG, X., AND ZHANG, H. Action-driven 3D Indoor Scene Evolution. *ACM SIGGRAPH Asia 35* (2016).

[99] MARTIN, S., THOMASZEWSKI, B., GRINSPUN, E., AND GROSS, M. Example-based Elastic Materials. *ACM TOG 30* (2011).

[100] MATTAUSCH, O., PANOZZO, D., MURA, C., SORKINE-HORNUNG, O., AND PAJAROLA, R. Object detection and classification from large-scale cluttered indoor scans. *CGF* (2014).

[101] NEWCOMBE, R. A., IZADI, S., HILLIGES, O., MOLYNEAUX, D., KIM, D., DAVISON, A., KOHLI, P., SHOTTON, J., HODGES, S., AND FITZGIBBON, A. KinectFusion: Real-time Dense Surface Mapping and Tracking. In *IEEE ISMAR* (2011).

[102] MEHRA, R., ZHOU, Q., LONG, J., SHEFFER, A., GOOCH, A., AND MITRA, N. J. Abstraction of Man-Made Shapes. *ACM SIGGRAPH Asia 28* (2009).

[103] MEHTA, D., RHODIN, H., CASAS, D., FUA, P., SOTNYCHENKO, O., XU, W., AND THEOBALT, C. Monocular 3D Human Pose Estimation In The Wild Using Improved CNN Supervision. In *3DV* (2017).

[104] MEHTA, D., SRIDHAR, S., SOTNYCHENKO, O., RHODIN, H., SHAFIEI, M., SEIDEL, H.-P., XU, W., CASAS, D., AND THEOBALT, C. VNect: Real-time 3D Human Pose Estimation with a Single RGB Camera. *ACM SIGGRAPH 36* (2017).

[105] MITRA, N. J., FLÖRY, S., OVSJANIKOV, M., GELFAND, N., GUIBAS, L., AND POTTMANN, H. Dynamic Geometry Registration. *SGP* (2007).

[106] MONSZPART, A., GUERRERO, P., CEYLAN, D., YUMER, E., AND MITRA, N. J. iMapper: Interaction-guided Scene Mapping from Monocular Videos. *ACM SIGGRAPH 38*, 4 (July 2019), 92:1–92:15.

[107] MONSZPART, A., MELLADO, N., BROSTOW, G., AND MITRA, N. J. RAPter: Rebuilding Man-made Scenes with Regular Arrangements of Planes. *ACM SIGGRAPH 34*, 4 (July 2015), 103:1–103:12.

[108] MONSZPART, A., THUEREY, N., AND MITRA, N. J. SMASH: Physics-guided Reconstruction of Collisions from Videos. *ACM SIGGRAPH Asia 35*, 6 (Nov. 2016), 199:1–199:14.

[109] MÜLLER, M., MCMILLAN, L., DORSEY, J., AND JAGNOW, R. Real-time Simulation of Deformation and Fracture of Stiff Materials. *Eurogr. Worksh. on Comp. Anim. and Sim.* (2001).

[110] NAN, L., SHARF, A., ZHANG, H., COHEN-OR, D., AND CHEN, B. SmartBoxes for Interactive Urban Reconstruction. In *ACM TOG* (2010).

[111] NAN, L., XIE, K., AND SHARF, A. A search-classify approach for cluttered indoor scene understanding. *ACM SIGGRAPH Asia 31* (2012).

[112] NEISSER, U. *Cognition and Reality: Principles and Implications of Cognitive Psychology*. W. H. Freeman, 1976.

[113] NEWCOMBE, R. A., FOX, D., AND SEITZ, S. M. DynamicFusion: Reconstruction and Tracking of Non-Rigid Scenes in Real-Time. In *IEEE CVPR* (2015).

[114] NEWELL, A., YANG, K., AND DENG, J. Stacked Hourglass Networks for Human Pose Estimation. In *ECCV* (2016).

[115] NI, K., JIN, H., AND DELLAERT, F. GroupSAC: Efficient Consensus in the Presence of Groupings. In *IEEE ICCV* (2009).

[116] NIESSNER, M., DAI, A., AND FISHER, M. Combining inertial navigation and ICP for real-time 3D surface reconstruction. *CGF Eurographics* (2014).

[117] NIESSNER, M., ZOLLHÖFER, M., IZADI, S., AND STAMMINGER, M. Real-time 3D Reconstruction at Scale using Voxel Hashing. *ACM SIGGRAPH Asia 32* (2013).

[118] OESAU, S., LAFARGE, F., AND ALLIEZ, P. Indoor scene reconstruction using feature sensitive primitive extraction and graph-cut. *ISPRS Journ. Photogramm. and Remote Sensing 90* (2014).

[119] O'SULLIVAN, C., DINGLIANA, J., GIANG, T., AND KAISER, M. K. Evaluating the Visual Fidelity of Physically Based Animations. *ACM TOG 22* (2003).

[120] PAN, Q., REITMAYR, G., ROSTEN, E., AND DRUMMOND, T. Rapid 3d modelling from live video. In *The 33rd International Convention MIPRO* (2010).

[121] PAULY, M., NILOY J. MITRA, GIESEN, J., GROSS, M., AND GUIBAS, L. Example-Based 3D Scan Completion. In *SGP* (2005).

[122] PHAM, T. T., CHIN, T.-J., SCHINDLER, K., AND SUTER, D. Interacting Geometric Priors For Robust Multimodel Fitting. *IEEE Transactions on Image Processing 23* (2014).

[123] PIRK, S., DIAMANTI, O., THIBERT, B., XU, D., AND GUIBAS, L. J. Shape-Aware Spatio-Temporal Descriptors for Interaction Classification. In *IEEE ICIP* (2017).

[124] PIRK, S., KRS, V., HU, K., RAJASEKARAN, S. D., KANG, H., YOSHIYASU, Y., BENES, B., AND GUIBAS, L. J. Understanding and Exploiting Object Interaction Landscapes. In *ACM SIGGRAPH Asia* (2017).

[125] POIRSON, P., AMMIRATO, P., FU, C.-Y., LIU, W., KOSECKÁ, J., AND BERG, A. C. Fast single shot detection and pose estimation. In *3DV* (2016).

[126] POPOVIĆ, J., SEITZ, S. M., ERDMANN, M., POPOVIĆ, Z., AND WITKIN, A. Interactive Manipulation of Rigid Body Simulations. *ACM SIGGRAPH* (2000).

[127] PURUSHWALKAM, S., GUPTA, A., KAUFMAN, D. M., AND RUSSELL, B. Bounce and Learn: Modeling Scene Dynamics with Real-World Bounces. In *ICLR* (2019).

[128] RAMALINGAM, S., AND BRAND, M. Lifting 3D Manhattan lines from a single image. In *IEEE ICCV* (2013).

[129] REN, S., HE, K., GIRSHICK, R., AND SUN, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *IEEE PAMI* (2017).

[130] ROGEZ, G., WEINZAEPFEL, P., AND SCHMID, C. LCR-Net++: Multi-person 2D and 3D Pose Detection in Natural Images. In *IEEE PAMI* (2019).

[131] ROTHER, C., KOHLI, P., FENG, W., AND JIA, J. Minimizing sparse higher order energy functions of discrete variables. In *IEEE CVPR* (2009).

[132] ROTHER, C., KOLMOGOROV, V., AND BLAKE, A. "GrabCut": interactive foreground extraction using iterated graph cuts. *ACM SIGGRAPH 23* (2004).

[133] RUSSELL, C., YU, R., AND AGAPITO, L. Video Pop-up: Monocular 3D Reconstruction of Dynamic Scenes. In *ECCV* (2014).

[134] RUSU, R. B., AND COUSINS, S. 3D is here: Point Cloud Library (PCL). In *IEEE ICRA* (2011).

[135] SALAS-MORENO, R. F., GLOCKEN, B., KELLY, P. H., AND DAVISON, A. J. Dense planar SLAM. In *IEEE ISMAR* (2014).

[136] SALZMANN, M., AND URTASUN, R. Physically-based Motion Models for 3D Tracking: A Convex Formulation. In *IEEE ICCV* (2011).

[137] SATKIN, S., AND HEBERT, M. 3DNN: Viewpoint invariant 3D geometry matching for scene understanding. In *IEEE CVPR* (2013).

[138] SAVVA, M., CHANG, A. X., HANRAHAN, P., FISHER, M., AND NIESSNER, M. SceneGrok: Inferring Action Maps in 3D Environments. *ACM SIGGRAPH Asia 33* (2014).

[139] SAVVA, M., CHANG, A. X., HANRAHAN, P., FISHER, M., AND NIESSNER, M. PiGraphs: Learning Interaction Snapshots from Observations. *ACM SIGGRAPH 35* (2016).

[140] SCHENK, O., AND GÄRTNER, K. Solving Unsymmetric Sparse Systems of Linear Equations with PARDISO. In *Future Gener. Comput. Syst.* (2004).

[141] SCHLECHT, J., AND BARNARD, K. Learning models of object structure. *NIPS* (2009).

[142] SCHNABEL, R., WAHL, R., AND KLEIN, R. Efficient RANSAC for Point-Cloud Shape Detection. In *CGF* (2007).

[143] SCHWING, A. G., FIDLER, S., POLLEFEYS, M., AND URTASUN, R. Box in the box: Joint 3D layout and object reasoning from single images. In *IEEE ICCV* (2013).

[144] SHAHROUDY, A., LIU, J., NG, T.-T., AND WANG, G. NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis. In *IEEE CVPR* (2016).

[145] SHAO, T., XU, W., ZHOU, K., WANG, J., LI, D., AND GUO, B. An interactive approach to semantic modeling of indoor scenes with an RGBD camera. In *ACM SIGGRAPH Asia* (2012).

[146] SHAO*, T., MONSZPART*, A., ZHENG, Y., KOO, B., XU, W., ZHOU, K., AND MITRA, N. J. Imagining the Unseen: Stability-based Cuboid Arrangements for Scene Understanding. *ACM SIGGRAPH Asia 33*, 6 (Nov. 2014), 209:1–209:11. * Joint first authors.

[147] SHARF, A., ALEXA, M., AND COHEN-OR, D. Context-based Surface Completion. *ACM SIGGRAPH 23* (2004).

[148] SHARF, A., HUANG, H., LIANG, C., ZHANG, J., CHEN, B., AND GONG, M. Mobility-Trees for Indoor Scenes Manipulation. In *CGF* (2013).

[149] SHEN, C.-H., FU, H., CHEN, K., AND HU, S.-M. Structure recovery by part assembly. *ACM SIGGRAPH Asia 31* (2012).

[150] SILBERMAN, N., AND FERGUS, R. Indoor scene segmentation using a structured light sensor. In *Proc. ICCV - Workshop on 3D Representation and Recognition* (2011).

[151] SILBERMAN, N., HOIEM, D., KOHLI, P., AND FERGUS, R. Indoor segmentation and support inference from RGBD images. *ECCV* (2012).

[152] SINHA, S. N., STEEDLY, D., SZELISKI, R., AGRAWALA, M., AND POLLEFEYS, M. Interactive 3D Architectural Modeling from Unordered Photo Collections. *ACM TOG 27* (2008).

[153] SMITH, B., KAUFMAN, D. M., VOUGA, E., TAMSTORF, R., AND GRINSPUN, E. Reflections on Simultaneous Impact. *ACM TOG 31* (2012).

[154] STRAUB, J., ROSMAN, G., FREIFELD, O., LEONARD, J. J., AND FISHER III, J. W. A Mixture of Manhattan Frames: Beyond the Manhattan World. In *IEEE CVPR* (2014).

[155] SU, J., SCHROEDER, C., AND FEDKIW, R. Energy Stability and Fracture for Frame Rate Rigid Body Simulations. *ACM SCA* (2009).

[156] TANG, D., NGO, J. T., AND MARKS, J. N-body Spacetime Constraints. *JVCA 6* (1995).

[157] TEH, C.-H., AND CHIN, R. T. On the Detection of Dominant Points on Digital Curves. *IEEE PAMI 11* (1989).

[158] TEKIN, B., ROZANTSEV, A., LEPETIT, V., AND FUA, P. Direct Prediction of 3D Body Poses from Motion Compensated Sequences. In *IEEE CVPR* (2016).

[159] TERZOPOULOS, D., PLATT, J., BARR, A., AND FLEISCHER, K. Elastically Deformable Models. *ACM SIGGRAPH 21* (1987).

[160] TOMÈ, D., RUSSELL, C., AND AGAPITO, L. Lifting from the Deep: Convolutional 3D Pose Estimation from a Single Image. In *IEEE CVPR* (2017).

[161] TOSHEV, A., AND SZEGEDY, C. DeepPose: Human Pose Estimation via Deep Neural Networks. In *IEEE CVPR* (2014).

[162] TULSIANI, S., GUPTA, S., FOUHEY, D., EFROS, A. A., AND MALIK, J. Factoring shape, pose, and layout from the 2D image of a 3D scene. In *IEEE CVPR* (2018).

[163] TULSIANI, S., SU, H., GUIBAS, L. J., EFROS, A. A., AND MALIK, J. Learning Shape Abstractions by Assembling Volumetric Primitives. In *IEEE CVPR* (2017).

[164] TWIGG, C. D., AND JAMES, D. L. Many-worlds Browsing for Control of Multibody Dynamics. *ACM TOG 26* (2007).

[165] UMETANI, N., IGARASHI, T., AND MITRA, N. J. Guided Exploration of Physically Valid Shapes for Furniture Design. *ACM SIGGRAPH 31* (2012).

[166] VICENTE, S., AND AGAPITO, L. Balloon Shapes: Reconstructing and Deforming Objects with Volume from Images. In *3DV* (2013).

[167] VON MARCARD, T., ROSENHAHN, B., BLACK, M. J., AND PONS-MOLL, G. Sparse Inertial Poser: Automatic 3D Human Pose Estimation from Sparse IMUs. *CGF Eurographics 36* (2017).

[168] WÄCHTER, A., AND BIEGLER, L. T. On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming. *Mathematical Programming 106* (2006).

[169] WANG, H., LIAO, M., ZHANG, Q., YANG, R., AND TURK, G. Physically Guided Liquid Surface Modeling from Videos. *ACM SIGGRAPH 28* (2009).

[170] WANG, H., AND O'SULLIVAN, C. Globally Continuous and Non-Markovian Crowd Activity Analysis from Videos. In *ECCV* (2016).

[171] WANG, K., SAVVA, M., CHANG, A. X., AND RITCHIE, D. Deep convolutional priors for indoor scene synthesis. In *ACM TOG* (2018).

[172] WATTERS, N., ZORAN, D., WEBER, T., BATTAGLIA, P., PASCANU, R., AND TACCHETTI, A. Visual Interaction Networks: Learning a Physics Simulator from Video. In *NIPS* (2017).

[173] WEI, P., ZHAO, Y., ZHENG, N., AND ZHU, S.-C. Modeling 4D Human-Object Interactions for Event and Object Recognition. In *IEEE ICCV* (2013).

[174] WEI, S.-E., RAMAKRISHNA, V., KANADE, T., AND SHEIKH, Y. Convolutional Pose Machines. In *IEEE CVPR* (2016).

[175] WEI, X., AND CHAI, J. VideoMocap: Modeling Physically Realistic Human Motion from Monocular Video Sequences. *ACM SIGGRAPH 29* (2010).

[176] WEI, X., ZHANG, P., AND CHAI, J. Accurate Realtime Full-body Motion Capture Using a Single Depth Camera. *ACM TOG 31* (2012).

[177] WHITING, E., OCHSENDORF, J., AND DURAND, F. Procedural Modeling of Structurally-sound Masonry Buildings. *ACM TOG 28* (2009).

[178] WITKIN, A., AND KASS, M. Spacetime Constraints. *ACM SIGGRAPH 22* (1988).

[179] XIAO, J., AND FURUKAWA, Y. Reconstructing the world's museums. In *IJCV* (2014).

[180] XIAO, J., RUSSELL, B., AND TORRALBA, A. Localizing 3D Cuboids in Single-View Images. In *NIPS* (2012).

[181] XIONG, X., AND HUBER, D. Using context to create semantic 3D models of indoor environments. *BMVC* (2010).

[182] XU, K., MA, R., ZHANG, H., ZHU, C., SHAMIR, A., COHEN-OR, D., AND HUANG, H. Organizing heterogeneous scene collections through contextual focal points. In *ACM SIGGRAPH* (2014).

[183] XU, Z., WU, H.-T., WANG, L., ZHENG, C., TONG, X., AND QI, Y. Dynamic Hair Capture Using Spacetime Optimization. *ACM TOG 33* (2014).

[184] YAN, F., SHARF, A., LIN, W., HUANG, H., AND CHEN, B. Proactive 3D Scanning of Inaccessible Parts. In *ACM SIGGRAPH* (2014).

[185] YAO, B., KHOSLA, A., AND FEI-FEI, L. Classifying Actions and Measuring Action Similarity by Modeling the Mutual Context of Objects and Human Poses. In *ICML* (2011).

[186] YEH, Y.-T., YANG, L., WATSON, M., GOODMAN, N. D., AND HANRAHAN, P. Synthesizing open worlds with constraints using locally annealed reversible jump MCMC. In *ACM SIGGRAPH* (2012).

[187] YUMER, E., AND KARA, L. B. Co-Abstraction of Shape Collections. *ACM SIGGRAPH Asia 31* (2012).

[188] ZHANG, H.-B., LEI, Q., ZHONG, B.-N., DU, J.-X., AND PENG, J. A Survey on Human Pose Estimation. In *Intelligent Automation and Soft Computing* (2016).

[189] ZHAO, X., HU, R., GUERRERO, P., MITRA, N., AND KOMURA, T. Relationship templates for creating scene variations. In *ACM SIGGRAPH Asia* (2016).

[190] ZHAO, X., WANG, H., AND KOMURA, T. Indexing 3D scenes using the interaction bisector surface. In *ACM TOG* (2014).

[191] ZHENG, B., ZHAO, Y., YU, J. C., IKEUCHI, K., AND ZHU, S.-C. Beyond Point Clouds: Scene Understanding by Reasoning Geometry and Physics. In *IEEE CVPR* (2013).

[192] ZHENG, Q., SHARF, A., WAN, G., LI, Y., MITRA, N. J., COHEN-OR, D., AND CHEN, B. Non-local scan consolidation for 3d urban scenes. *ACM SIGGRAPH 29* (2010).

[193] ZHENG, Y., CHEN, X., CHENG, M.-M., ZHOU, K., HU, S.-M., AND MITRA, N. J. Interactive Images: Cuboid Proxies for Smart Image Manipulation. *ACM SIGGRAPH 31* (2012).

[194] ZHOU, X., ZHU, M., LEONARDOS, S., DERPANIS, K., AND DANIILIDIS, K. Sparseness Meets Deepness: 3D Human Pose Estimation from Monocular Video. In *IEEE CVPR* (2016).

[195] ZISSERMAN, M. A. A. Robust object tracking. In *ACCV* (1995).

[196] ZIVKOVIC, Z. Improved Adaptive Gaussian Mixture Model for Background Subtraction. *IEEE ICPR 2* (2004).

[197] ZOLLHÖFER, M., THIES, J., GARRIDO, P., BRADLEY, D., BEELER, T., PÉREZ, P., STAMMINGER, M., NIESSNER, M., AND THEOBALT, C. State of the Art on Monocular 3D Face Reconstruction, Tracking, and Applications. *CGF 37* (2018).

[198] ZOU, C., GUO, R., LI, Z., AND HOIEM, D. Complete 3D Scene Parsing from an RGBD Image. *IJCV 127* (2019).