# Data-driven Methods for Interactive Visual Content Creation and Manipulation

Dissertation zur Erlangung des Grades des
Doktors der Ingenieurwissenschaften der
Naturwissenschaftlich-Technischen Fakultäten der
Universität des Saarlandes

Vorgelegt durch

Arjun Jain
Max-Planck-Institut Informatik
Campus E1 4
66123 Saarbrücken
Germany

am 4. February 2013 in Saarbrücken

**Dekan – Dean**
Prof. Dr. Markus Bläser, Universität des Saarlandes, Saarbrücken, Germany

**Kolloquium – Examination**

Datum – Date:
19. March 2014

Vorsitzender – Chair:
Prof. Dr. Sebastian Hack, Universität des Saarlandes, Saarbrücken, Germany

Prüfer – Examiners:
Prof. Dr. Hans-Peter Seidel, Universität des Saarlandes, Saarbrücken, Germany
Prof. Dr. Thorsten Thormählen, Philipps-Universität Marburg, Marburg, Germany
Prof. Dr. Bernt Schiele, Universität des Saarlandes, Saarbrücken, Germany

Protokoll – Reporter:
Dr. Klaus Hildebrandt

# Abstract

Software tools for creating and manipulating visual content — be they for images, video or 3D models — are often difficult to use and involve a lot of manual interaction at several stages of the process. Coupled with long processing and acquisition times, content production is rather costly and poses a potential barrier to many applications. Although cameras now allow anyone to easily capture photos and video, tools for manipulating such media demand both artistic talent and technical expertise. However, at the same time, vast corpuses with existing visual content such as Flickr, YouTube or Google 3D Warehouse are now available and easily accessible.

This thesis proposes a *data-driven* approach to tackle the above mentioned problems encountered in content generation. To this end, statistical models trained on semantic knowledge harvested from existing visual content corpuses are created. Using these models, we then develop tools which are easy to learn and use, even by novice users, but still produce high-quality content. These tools have intuitive interfaces, and enable the user to have precise and flexible control. Specifically, we apply our models to create tools to simplify the tasks of video manipulation, 3D modeling and material assignment to 3D objects.

# Kurzfassung

Softwarewerkzeuge zum Erstellen und Bearbeiten von visuellen Inhalten – seien es Bilder, Videos oder 3D-Modelle – sind häufig schwierig zu bedienen und erfordern viel manuelle Interaktion an verschiedenen Stellen des Verfahrens. In Verbindung mit langen Bearbeitungs- und Erfassungszeiten ist die Erzeugung von Inhalten eher aufwendig und stellt ein potentielles Hindernis für viele Anwendungen dar. Obwohl heute Kameras jedem Anwender auf einfache Art und Weise erlauben Bilder und Videos aufzunehmen, erfordern Werkzeuge zur Bearbeitung dieser sowohl künstlerisches Talent, als auch technische Kompetenz. Gleichzeitig sind riesige Korpora mit bereits vorhandenen visuellen Inhalten, wie zum Beispiel Flickr, Youtube oder Google 3D Warehouse, verfügbar und leicht zugänglich.

Diese Arbeit stellt einen *datengetriebenen* Ansatz vor, der die erwähnten Probleme der Inhaltserzeugung behandelt. Zu diesem Zweck werden statistische Modelle erzeugt, die auf semantischem Wissen trainiert worden sind, welches aus bestehenden Korpora von visuellen Inhalten gesammelt worden ist. Durch die Verwendung dieser Modelle ist es möglich Werkzeuge zu entwickeln, die sogar von unerfahrenen Anwendern einfach zu erlernen und zu benutzen sind, aber dennoch qualitativ hochwertige Inhalte produzieren. Diese Werkzeuge haben intuitive Benutzeroberflächen und geben dem Benutzer eine präzise und flexible Kontrolle. Insbesondere werden die Modelle eingesetzt, um Werkzeuge zu erzeugen, die Aufgaben Videobearbeitung, 3D-Modellerstellung und Materialzuweisung zu 3D-Modellen vereinfachen.

# Summary

Software tools for creating and manipulating visual content are often difficult to use and involve a lot of manual interaction at several stages of the process. Coupled with long processing and acquisition times, content production is rather costly and poses a potential barrier to many applications.

The challenge undertaken in this work is to design data-driven algorithms that optimize two conflicting constraints : *a*) being able to produce high-quality content, but also are intuitive and easy to learn and use even by novice users and *b*) being (semi-)automatic while still enabling the user to have precise and flexible control.

We will present three interactive systems for visual content creation. These methods have a common goal to assist the user in the task of content production in an interactive setting, and they all use a suitable data-driven approach. Semantic information is gathered by training on data from corpuses such as Flickr or Google 3D Warehouse. With this information, our algorithms are employed to create tools for intuitive and high-quality visual content creation and manipulation.

First, a video manipulation tool is presented which enables the user to edit the shape of human actors in video using intuitive semantic sliders. Edits of this sort in video were not possible before our method. Also, state-of-the-art methods for human detection and articulated pose estimation can benefit from being able to make such manipulations to video as they require a lot of training data. Manually collected datasets do exist, but since a person can be edited in images or video using our approach, we can now control the captured variations w.r.t. appearance, shape and pose, and thus generate on-demand training data. Next, a data-driven 3D modeling approach is explained which simplifies the task of 3D modeling through rapid instantiation of new models by blending between two existing database shapes. The user can choose how many parts need to be replaced simply by moving a slider. Finally, we propose a method for automatic material suggestions for 3D objects. We fully automatically assign plausible material parameters to 3D objects, and further propose a novel user interface that provides ranked material suggestions.

**Tracking and Reshaping of Humans in Videos** We present a system for quick and easy manipulation of the body shape and proportions of a human actor in arbitrary video footage. The approach is based on a morphable model of 3D human

shape and pose that was learned from laser scans of real people. The algorithm commences by spatio-temporally fitting the pose and shape of this model to the actor in either single-view or multi-view video footage. Once the model has been fitted, semantically meaningful attributes of body shape, such as height, weight or waist girth, can be interactively modified by the user. The changed proportions of the virtual human model are then applied to the actor in all video frames by performing an image-based warping. By this means, we can now conveniently perform spatio-temporal reshaping of human actors in video footage which we show on a variety of video sequences.

We use methods from this approach to extend existing training sets with explicit variation control. In this work we are interested in the problem of articulated people detection and pose estimation in real-world sports scenes. State-of-the-art methods for human detection and pose estimation require many training samples for best performance. While large, manually collected datasets exist, the captured variations w.r.t. appearance, shape and pose are often uncontrolled, thus limiting the overall performance. In order to overcome this limitation we propose a new technique to extend an existing training set that allows explicit control of pose and shape variations. To do this we build on recent advances in computer graphics to generate samples with realistic appearance and background while modifying body shape and pose.

**Exploring Shape Variations by 3D-Model Decomposition and Part-based Recombination** In this work, we present a system that allows new shapes to be created by blending between shapes taken from a database. We treat the shape as a composition of parts; blending is performed by recombining parts from different shapes according to constraints deduced by shape analysis. The analysis involves shape segmentation, contact analysis, and symmetry detection. The system can be used to rapidly instantiate new models that have similar symmetry and adjacency structure to the database shapes, yet vary in appearance.

**Automatic Material Suggestions for 3D Objects** The material found on 3D objects and their parts in our everyday surroundings is highly correlated with the geometric shape of the parts and their relation to other parts of the same object. We propose to model this context-dependent correlation by learning it from a database containing several hundreds of objects and their materials. Given a part-based 3D object without materials, the learned model can be used to fully automatically assign plausible material parameters, including diffuse color, specularity, gloss, and transparency. Further, we propose a user interface that provides material suggestions. This user-interface can be used, for example, to refine the automatic suggestion. Once a refinement has been made, the model incorporates this information, and the automatic assignment is incrementally improved. Results are given for objects with different numbers of parts and with different topological complexity. A user

study confirms that our method significantly simplifies and accelerates the material assignment task compared to other approaches.

# Zusammenfassung

Softwarewerkzeuge zum Erstellen und Bearbeiten von visuellen Inhalten sind häufig schwierig zu bedienen und erfordern viel manuelle Interaktion an verschiedenen Stellen des Verfahrens. In Verbindung mit langen Bearbeitungs- und Erfassungszeiten ist die Erzeugung von Inhalten eher aufwendig und stellt ein potentielles Hindernis für viele Anwendungen dar.

Die Herausforderung in dieser Arbeit liegt darin, datengetriebene Algorithmen zu entwerfen, welche die zwei folgenden widersprüchlichen Bedingungen optimieren: (*a*) Sie sollen in der Lage sein, qualitativ hochwertigen Inhalt zu generieren, aber gleichzeitig intuitiv und von unerfahrenen Anwendern einfach zu erlernen und zu benutzen zu sein. (*b*) Es sollen (semi-)automatische Algorithmen verwendet werden und dem Benutzer trotzdem eine präzise und flexible Kontrolle zu geben.

In dieser Arbeit werden drei interaktive Systeme für die Erstellung von visuellem Inhalt vorstelst. Diese Verfahren haben das gemeinsame Ziel dem Anwender bei der Aufgabe der Inhaltserzeugung in einer interaktiven Umgebung zu unterstützen, wozu sie einen passenden datengetriebenen Ansatz verwenden. Semantische Informationen werden durch Trainieren auf Daten von Korpora, wie zum Beispiel Flickr oder Google 3D Warehouse, gesammelt. Mit diesen Informationen werden unsere Algorithmen verwendet, um Werkzeuge für eine intuitive und hochwertige Inhaltserzeugung und -bearbeitung zu entwickeln.

Zunächst wird ein Videobearbeitungswerkzeug vorgestellt, das es dem Benutzer erlaubt, die Gestalt menschlicher Akteure in Videos mit Hilfe von intuitiven semantischen Schiebereglern zu verändern. Änderungen dieser Art in Videos waren vorher nicht möglich. Außerdem können Verfahren zur Erkennung von Personen und Schätzung ihrer Körperhaltung von den Bearbeitungsmöglichkeiten profitieren, da sie eine große Menge an Trainingsdaten benötigen. Manuell gesammelte Datensätze existieren zwar, da aber eine Person mit unserem Verfahren in Bildern und Videos bearbeitet werden kann, sind wir jetzt auch in der Lage die aufgenommenen Variationen bezüglich Erscheinung, Gestalt und Haltung zu kontrollieren und folglich neue Trainingsdaten auf Anfrage zu generieren. Danach wird ein datengetriebenes 3D-Modellierungsverfahren erklärt, welches die Aufgabe der 3D-Modellierung vereinfacht, indem neue Modelle durch Mischen aus zwei bestehenden Datenbankmodellen schnell instanziiert werden. Schließlich, präsentieren wir ein Verfahren

zum automatisierten Vorschlagen von Materialen von 3D-Objekten. Wir weisen 3D-Objekten plausible Materialenparameter vollautomatisch zu und stellen eine neue Benutzeroberfläche vor, die eine sortierte Liste von Materialvorschlägen anbietet.

**Tracking und Umformung von Menschen in Videos** Wir präsentieren ein System zur schnellen und einfachen Bearbeitung der Körperform und der Körperproportionen eines Akteurs in beliebigen Videoaufnahmen. Der Ansatz basiert auf einem verformbaren Model der 3D-Körperform und -haltung eines Menschen, das von Laserscans realer Personen gelernt worden ist. Der Algorithmus beginnt mit dem raum-zeitlichen Anpassen der Form und Haltung des Models an den Akteur in einem oder mehreren Blickwinkeln der Videoaufnahme. Nachdem das Model angpasst worden ist, können semantisch bedeutende Merkmale der Körperform, wie etwa Größe, Gewicht oder Taillenumfang interaktiv verändert werden. Die veränderten Proportionen des virtuellen menschlichen Modells werden dann auf den Akteur in allen Einzelbildern des Videos durch eine bildbasierte Deformation angewendet.

Wir verwenden Verfahren aus diesem Ansatz um existierende Trainingssätze um eine explizite Kontrolle in der Variationen zu erweitern. In dieser Arbeit sind wir an dem Problem der Detektion von Menschen und der Schätzung ihrer Körperhaltung in realen Sportszenen interessiert. Aktuelel Verfahren zur Detektion von Menschen und der Schätzung der Körperhaltung benötigen viele Trainingsbeispiele um beste Ergebnisse zu erzielen.

Obwohl manuell gesammelte Datensätze existieren, sind die aufgenommenen Variationen bezüglich Erscheinung, Gestalt und Haltung unkontrolliert und limitieren dadurch die allgemeine Leistungsfähigkeit der Algorithmen. Um diese Einschränkung zu überwinden, schlagen wir einen neuen Ansatz vor, der vorhandene Trainingssätze erweitert und eine explizite Kontrolle über die Variationen in Haltung und Form erlaubt. Dafür bauen wir auf aktuellen Fortschritten der Computergrafik auf, um Beispiele mit realistischer Erscheinung und realistischem Hintergrund zu generieren, während wir die Körperform und -haltung modifizieren.

**Erforschen von Modellvariationen durch Zerlegung von 3D-Modellen und teilebasierter Rekombination** In dieser Arbeit stellen wir ein System vor, dass es erlaubt, neue Modelle durch Mischen von Modellen aus einer Datenbank zu erzeugen. Wir behandeln das Modell als eine Komposition von Einzelteilen. Das Mischen findet statt, indem Teile von verschiedenen Modellen nach Bedingungen rekombiniert werden, die aus einer Modellanalyse hergeleitet worden sind. Die Analyse beinhaltet Modellsegmentierung, Kontaktanalyse und Symmetriedetektion. Das System kann dazu benutzt werden, schnell neue Modelle zu instantiieren, die ähnliche Symmetrie- und Nachbarschaftsstrukturen zu vorhandenen Datenbankmodellen ausweisen, jedoch in ihrer Erscheinung variieren.

**Automatische Materialvorschläge für 3D-Objekte** Die Materialien, die wir bei 3D-Objekten und ihren Teilen aus unserer täglichen Umgebung finden, ist stark korreliert zu der geometrischen Form der Teile und ihrer Relation zu anderen Teilen des Objekts. Wir schlagen vor, diese kontextabhängige Korrelation zu modellieren, indem man sie aus einer Datenbank bestehend aus mehreren hundert Objekten und ihren Materialien lernt. Gegeben ein teilebasiertes 3D-Objekt ohne Materialien, kann das gelernte Modell dazu benutzt werden, plausible Materialien, einschließlich diffuser Farbe, Spiegelung, Glanz und Transparenz, vollautomatisch zuzuweisen. Weiterhin stellen wir eine Benutzeroberfläche vor, die Materialvorschläge anbietet. Diese Oberfläche kann benutzt werden, um den automatischen Vorschlag beispielsweise zu verfeinern. Nach dem Verfeinern berücksichtigt das Modell diese Informationen und die automatische Zuweisung wird verbessert. Ergebnisse werden für verschiedene Objekte mit einer unterschiedlichen Anzahl an Teilen und mit unterschiedlicher topologischer Komplexität gezeigt. Eine Nutzerstudie bestätigt, dass unser Verfahren verglichen mit anderen Ansätzen die Materialzuweisung signifikant vereinfacht und verbessert.

# Contents

# 1 Introduction

In this thesis, we propose several novel data-driven models that can be utilized to synthesize and manipulate visual content. In this introductory chapter, we motivate our research, present our main contributions, and outline the rest of the thesis.

## 1.1 Motivation

It has been observed [Florida 2003] that the demand for richer, more engaging visual content is constantly increasing. This development is further driven by recent growth in the capabilities of consumer-grade hardware such as GPU-based computers, multi-core cell processor based gaming devices (such as Microsoft XBox® and Sony PlayStation®), depth sensors (e.g. Microsoft Kinect®) and smart mobile phones and algorithms that can exploit their capabilities. However, creating high-quality content is challenging and requires specialized skills, knowledge, and training and remains accessible only to experts. It is a non-trivial task to edit a video, or create a 3D model. At the same time, the availability and accessibility of visual content has become ubiquitous. Huge visual content corpuses such as Flickr, YouTube or Google 3D Warehouse exist and this easily available visual data is continually increasing. 72 hours of video are uploaded per minute on YouTube. 2.5 billion photos are uploaded to Facebook each month. On Google SketchUp's 3D Warehouse, one can find incredibly detailed 3D models of most major building structures of the world. In this thesis, we propose algorithms to learn statistical models from semantic information extracted from existing artist-created content, and then use these learned models in the content creation workflow, such that new content can be created automatically. These tools also help ease some tedious manual work in the content creation pipeline and thus enable the designer to concentrate on artistic creativity and styling.

*The premise of this dissertation is that statistical models learned from existing visual content databases can be used for automatizing content creation and to design intuitive interfaces which are easier to learn and use even for novice users.*

## 1.2 Contributions

This section lists individual contributions made in the this thesis.

In Chapter 4, the first video manipulation system which enables semantically-based edits of human actors in video streams is introduced. Using this system, actors in existing video footage can also be made to look e. g., taller, thinner, more muscular, etc. The main contributions to data-driven video manipulation in Chapter 4 (published as [Jain et. al. 2010]) are:

- An end-to-end system for quick and easy manipulation of the body shape and proportions of a human actor in arbitrary video footage.
- Use of a morphable human shape and pose model, learned from a database of 3D laser scans of humans, with semantically control parameters.
- An algorithm for spatio-temporally fitting the pose and shape of this model to the actor in either single-view or multi-view video footage.

In computer vision, state-of-the-art methods for human detection and pose estimation from images require many training samples for best performance. While large manually-collected datasets exist, the captured variations w.r.t. appearance, shape and pose are often uncontrolled, thus limiting the overall performance. In order to overcome this limitation we employ techniques from *MovieReshape*, presented in Chapter 4, to extend an existing training set that allows explicit control of pose and shape variations. The main contributions in this regard are:

- A method for automatic generation of multiple training examples from an arbitrary set of images with annotated human body poses, with full control over the shape and pose variations.
- An evaluation of our data generation method on the task of articulated human detection and pose estimation. Significant improvement in performance is found when the training sets are extended with the automatically-generated images.
- A joint model that directly integrates evidence from an appropriately trained deformable part model (DPM, [Felzenszwalb et al. 2010]) into a pictorial structures framework and an evaluation of how this joint model further improves performance.

- A new challenge of joint detection and pose estimation of multiple articulated people in challenging real-world scenes.

In this joint work, my contribution was designing the part of the pipeline used for generating novel training sets (Section 4.7) by changing the shape and pose of humans in images.

In Chapter 5, we present a tool which advances the field of data-driven 3D modeling which enables exploration of shape variations by 3D-model decomposition and part-based recombination. The main contributions of Chapter 5 (published as [Jain et. al. 2011]) are:

- A tool to produce custom detailed 3D models by recombining parts from different shapes present in a database, according to constraints deduced by shape analysis.
- An algorithm for systematic hierarchical shape analysis of 3D models involving shape segmentation, part contact analysis, and symmetry detection.

Chapter 6 (published as [Jain et. al. 2012]) proposes the first ever approach of its kind to computationally model the relation of shape and material by learning it from a database of multi-component 3D objects with materials. This model can then be used to automatically assign materials to 3D objects or can be employed in a user-interface to provide a ranked list of the most likely materials. Specific contributions made in this work are as follows:

- A model of the relation between materials and shape as well as context, called the *material memex*.
- Automatic assignment of materials using this model.
- A novel interface to guide a user when assigning materials by providing ranked material suggestions.
- A user study of task performance when using conventional slider or text interfaces compared to our interface.

A full list of the author's related publications is found on Page I of the Appendix.

## 1.3 Outline

This thesis is structured as follows. After this introduction, a background on processing databases of 3D meshes is given in Chapter 2, before we review previous work in Chapter 3. From Chapter 4 to Chapter 6, novel techniques are presented in detail. The thesis is completed by a conclusion in Chapter 7 which also contains a discussion of future work.

# 2

# Background

Figure 2.1: A subset of models from a 3D model collection.

In this chapter we provide an overview of steps employed for processing databases of 3D meshes, which is a prerequisite for the algorithms presented in Chapter 4–6. We first introduce the special case when complete congruence over the mesh collections can be established (Section 2.1). This model is later used in a video manipulation application introduced in Chapter 4. A preprocessing pipe-line for heterogeneous 3D model collections (Figure 2.1) is then explained. Object are decomposed into parts, and a graph encoding these part-relationships is presented in Section 2.2. This machinery is then later utilized for the automatic shape modeling tool of Chapter 5 and the automatic material assignment tool of Chapter 4.

## 2.1 Full Mesh Correspondence

A collection of shapes with assigned correspondences implicitly encodes a deformation model for the collection. Such shapes can be seen as high-dimensional points within a common coordinate system, and their principal modes of variations can be directly extracted using statistical tools. For example, [Blanz and Vetter 1999] in their highly influential work explore this idea in the context of 3D face models. A similar framework has been used to analyze shapes of human bodies in consistent poses [Allen, Curless and Popović 2003].

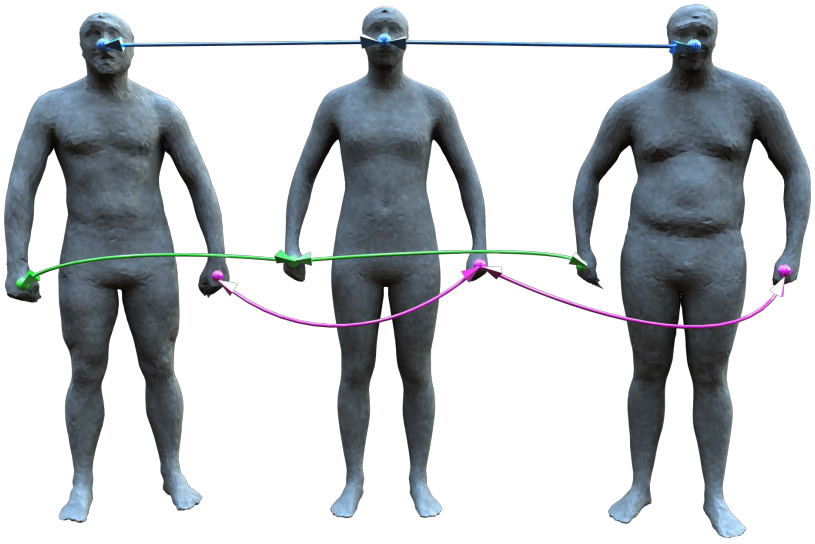


**Figure 2.2:** One-to-one correspondence between mesh vertices across models from the human shapes collection is illustrated. Such a correspondence is established for all mesh vertices across all exemplars.

Such a morphable model of a 3D face, or a human body is based on a collection of 3D faces $\mathcal{C}_{face}$ or human bodies $\mathcal{C}_{body}$ respectively. Morphing between them requires full correspondence between any two meshes across all exemplars. A semi-supervised variant of the ICP algorithm is used for computing correspondence between the exemplar models.

### 2.1.1 Mesh Representation

Let us call the 3D object (e.g. a face or a human body) $O_i \in \mathcal{C}$ where the set of $n^s$ such exemplar objects $O_i$ is called a collection $\mathcal{C} = \{O_i \mid i = 1,2,\ldots,n^s\}$. Let us denote the geometry of $O_i$ by a shape-vector $\mathbf{S} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n)^T \in \mathbb{R}^{3n}$ where $\mathbf{v}_i \in \mathbb{R}^3$ contains the $X, Y$ and $Z$ coordinates of its $n$ vertices.

Let us denote the average geometry of collection $\mathcal{C}$ as $\bar{\mathbf{S}} = \frac{1}{n^s}\sum_{i=1}^{n^s} \mathbf{S}_i$. Each exemplar differs from the average by the vector $\Delta\mathbf{S}_i = \mathbf{S}_i - \bar{\mathbf{S}}$. An example collection $\mathcal{C}_{body}$

**Table 2.1:** Table of symbols.

| Entity | Symbol |
|---|---|
| A collection of 3D objects | $\mathcal{C} = \{O_i \mid i = 1, 2, \ldots, n^s\}$ |
| An object in collection $\mathcal{C}$ | $O \in \mathcal{C}$ |
| Number of objects in collection $\mathcal{C}$ | $n^s \in \mathbb{N}$ |
| 3D position of $i^{th}$ mesh vertex of object $O$ | $(X_i, Y_i, Z_i) \in \mathbb{R}^3$ |
| No. of 3D points in geometry of object $O$ | $n \in \mathbb{N}$ |
| A 3D vertex in the geometry of object $O$ | $\mathbf{v}_i \in \mathbb{R}^3$ |
| Vector describing the geometry of object $O$ | $\mathbf{S} = (\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n)^T \in \mathbb{R}^{3n}$ |
| Average geometry of collection $\mathcal{C}$ | $\bar{\mathbf{S}} \in \mathbb{R}^{3n}$ |
| Difference in geometry of $O_i$ from $\bar{\mathbf{S}}$ | $\Delta\mathbf{S}_i = \mathbf{S}_i - \bar{\mathbf{S}} \in \mathbb{R}^{3n}$. |
| Data-matrix of shape-vectors with zero mean | $\mathsf{A} = [\Delta\mathbf{S}_1, \ldots, \Delta\mathbf{S}_{n^s}] \in \mathbb{R}^{3n \times n^s}$ |
| Co-variance matrix | $\mathsf{C} = \mathsf{A}\mathsf{A}^T \in \mathbb{R}^{3n \times 3n}$ |
| The $i^{th}$ eigenvector of $\mathsf{C}$ | $\mathbf{s}_i \in \mathbb{R}^{3n}$ |
| The eigenvalue corresponding to $\mathbf{s}_i$ of $\mathsf{C}$ | $\alpha_i \in \mathbb{R}$ |
| No. of basis vectors in reduced PCA space | $M \in \mathbb{N}$ |
| Geometry parameters of the final model | $\mathbf{\Lambda} \in \mathbb{R}^{3M}$ |
| Geometry parameter for the $i^{th}$ eigen-vector $\mathbf{s}_i$ | $\lambda_i \in \mathbb{R}, \mathbf{\Lambda} = \{\lambda_i \mid i = 1, \ldots, 3M\}$ |
| No. of real-valued semantic attributes of $O_i$ | $L \in \mathbb{Z}^+$ |
| The $i^{th}$ real-valued semantic attribute | $f_i \in \mathbb{R}$ |
| Linear map from semantic to PCA dimensions | $\mathsf{M} \in \mathbb{R}^{M-1 \times L+1}$ , $\mathsf{M}[f_1, \ldots, f_L, 1] = \mathbf{\Lambda}$ |

of 3D human bodies is shown in Figure 2.3, with the average $\bar{\mathbf{S}}_{body}$ model in red. Please refer to Table 2.1 for the symbols used.

### 2.1.2 PCA Analysis

This set containing all $\Delta\mathbf{S}_i$ can then be subjected to principal component analysis, which seeks a set of $n^s$ orthonormal vectors, $\mathbf{s}_i$. The $i$ th vector, $\mathbf{s}_i$, is chosen such that

$$\alpha_i = \frac{1}{n^s} \sum_{k=1}^{n^s} (\mathbf{s}_i^T \Delta\mathbf{S}_k)^2$$

is a maximum, subject to the orthonormal constraint

$$\mathbf{s}_j^T \mathbf{s}_i = \begin{cases} 1, & \text{if } j = i \\ 0, & \text{otherwise} \end{cases}$$

where vectors $\mathbf{s}_i$ and scalars $\alpha_i$ are eigenvectors and eigenvalues, respectively, of the covariance matrix $\mathsf{C} = \frac{1}{n^s} \sum_{i=1}^{n^s} \Delta\mathbf{S}_i \Delta\mathbf{S}_i^T = \mathsf{A}\mathsf{A}^T$, where the matrix $\mathsf{A} = [\Delta\mathbf{S}_1, \ldots, \Delta\mathbf{S}_{n^s}]$.

**Figure 2.3:** Database of human models with average model shown in red.

One can project the data in a lesser $M \ll n^s$ dimensional space by discarding eigenvectors corresponding to eigenvalues below a certain threshold. Taking any linear combination in the reduced PCA space, we can create an unlimited number of new samples which are linear combinations of exemplars of the collection. If we represent the geometry parameters by $\lambda_i \in \mathbf{\Lambda}$ where $\lambda_i$ is the coefficient corresponding to the $i$-th eigenvector $\mathbf{s}_i$, the generative deformation model of the collection can be represented as

$$\mathbf{S}_{model} = \bar{\mathbf{S}} + \sum_{k=1}^{M} \lambda_i \mathbf{s}_i$$

### 2.1.3 Morphable Model with Semantic Attributes

While PCA helps to characterize the space of variation, the PCA space parameters do not correspond to semantically meaningful dimensions. The modification of a single PCA parameter $\lambda_i$ will simultaneously modify a combination of shape aspects that we find intuitively plausible, e.g. for a body, parameters such as weight or strength of muscles. If each exemplar object $O_i$ also has $L$ real-valued semantic attributes associated to it (as seen in Figure 2.4), one could learn a linear mapping $\mathsf{M} \in \mathcal{M}\ (M-1) \times (L+1))$ between the $L$ semantic dimensions and the $m$ reduced PCA space dimensions $\mathsf{M}[f_1, \ldots, f_L, 1] = \mathbf{\Lambda}$, where $f_i$ are semantic attribute values, and $\mathbf{\Lambda}$ are corresponding PCA space coefficients. This mapping $\mathsf{M}$ can be learnt simply aggregating all semantic feature vectors into a matrix $\mathsf{F}$, and all reduced PCA vectors into another matrix $\mathsf{L}$, and solving $\mathsf{M} = \mathsf{L}\mathsf{F}^{+}$, where $\mathsf{F}^{+}$ is the pseudoinverse of $\mathsf{F}$.

While such methods can indeed learn useful modes of variation for specific classes of shapes, their reliance on accurate correspondences is a significant limitation for the task of exploring unorganized collections of 3D shapes. In particular, there

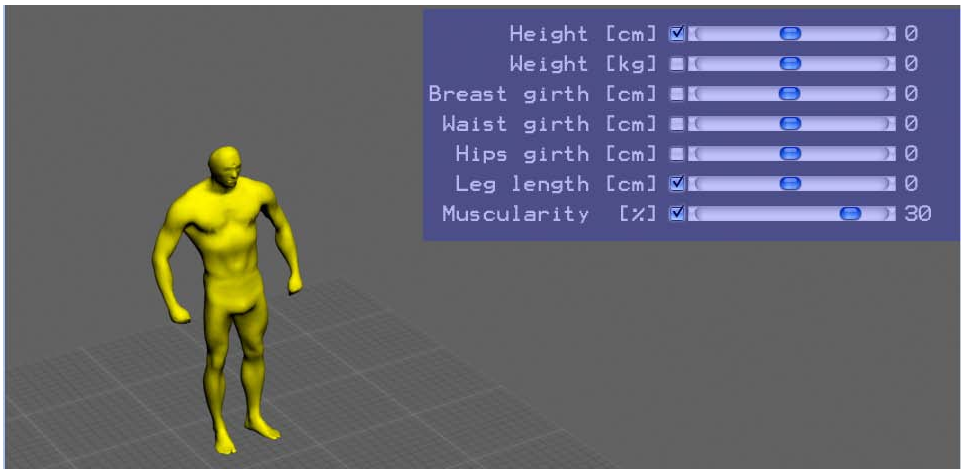



**Figure 2.4:** Various semantic attributes such as *height*, *weight* and *muscularity* can be used to navigate in the space spanned by the exemplar human body models.

is typically a large amount of variation in topology and geometric quality across models in public repositories, even those within the same class. In the face of such variations, global correspondence detection remains a challenging open problem (see [van Kaick et al. 2011]).

## 2.2 Graph Representation of Part Relationships

Statistical analysis over part-based object collections is not as straight forward as collections with full mesh correspondence (Section 2.1) primarily due to the difficulty of establishing correspondences across such collections of shapes, both at the level of each individual component and at the level of the complete shape. We look at statistical analysis over such collections of part-based objects in Chapter 5–6.

As noted by [Xu et al. 2010] it is not uncommon to see point sets, polygon soups, and water-tight meshes all within a single collection of models. Thus, before we study the principal modes of variations for such shape collections, we first pre-process all 3D objects from the collection in a **Shape Analysis** step. For both techniques presented in Chapter 5–6, we use the same pre-processing machinery as explained below.

### Shape Analysis

Shape analysis is used to find the relations between parts that constitute a shape. We start from $\mathcal{S} := \{S_i | O_i \in \mathcal{C}, i = 1, \ldots, n^s\}$, where $\mathcal{C}$ is the set of $n^s$ shapes $S_i$ in the database. Each shape $S_i$ is represented as a polygonal mesh. Our database comprises of different man-made objects, providing no symmetry or hierarchy information,

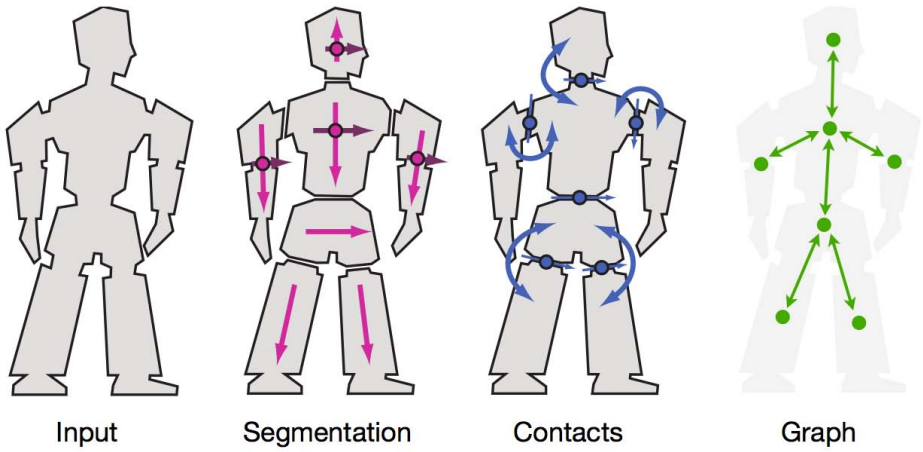



**Figure 2.5:** Analysis of a database of many 3D objects leading to a representation of shape-part relationships based on contacts between parts.

which are taken from 3D model repositories on the Internet. These models typically have different scales and inconsistent alignment to the global coordinate axes (as is typically the case for 3D models from Internet repositories).

This analysis is run on every shape in the database independently. It consists of point sampling, segmentation and contact analysis (see Figure 2.5).

### 2.2.1 Segmentation

The $i$-th shape $S_i$ is decomposed into $n_i^{\mathrm{p}}$ parts $S_i = \bigcup_{j=1}^{j=n_i^{\mathrm{p}}} P_{i,j}$ which are again polygonal meshes. In our case, segments are connected components of the polygonal input mesh, which are generated by region growing. Region growing is a bottom up method, where an initial set of 3D vertices are iteratively merged according to similarity constraints. We start by choosing an arbitrary seed vertex and comparing it with its neighboring vertices. A region is grown from the seed vertex by adding in neighboring vertices that have the same material, and are connected along the mesh. When the growth of one region stops we simply choose another seed vertex which does not yet belong to any region and start again. This whole process is continued until all vertices belong to some region.

### 2.2.2 Point Cloud Sampling

Next, every part $P_{i,j}$ is re-sampled to a point cloud $\bar{P}_{i,j}$ for further processing. We do this for two reasons: *i*) it is not uncommon to see point sets, polygon soups, models with broken mesh triangulation, etc all within a single collection of models and *ii*)

**Table 2.2:** Table of symbols.

| Entity | Symbol |
|---|---|
| The shape of object $O_i$ | $S_i \in \mathcal{S}$ |
| Set of shapes in the collection $C$ | $\mathcal{S} := \{S_i \mid O_i \in \mathcal{C}, i = 1, \ldots, n^{\mathrm{s}}\}$ |
| No of objects in collection | $n^{\mathrm{s}} \in \mathbb{N}$ |
| No of parts of shape $S_i$ | $n_i^{\mathrm{p}} \in \mathbb{N}$ |
| $j^{th}$ part of a shape $S_i$ | $P_{i,j} \mid S_i = \bigcup_{j=1}^{j=n_i^{\mathrm{p}}} P_{i,j}$ |
| Point cloud of part $P_{i,j}$ | $\bar{P}_{i,j}$ |
| Global to local scaling matrix for $P_{i,j}$ | $\mathsf{S}_{i,j} \in \mathbb{R}^{3\times3}$ |
| Global to local rotation matrix for $P_{i,j}$ | $\mathsf{R}_{i,j} \in \mathbb{R}^{3\times3}$ |
| Global to local coordinate transform for $P_{i,j}$ | $\mathsf{T}_{i,j} \mid \mathsf{T}_{i,j} = \mathsf{S}_{i,j}\mathsf{R}_{i,j}$ |
| A point of a part $P_{i,j}$ in global coordinates | $\mathbf{p} \in \mathbb{R}^3$ |
| A point of a part $P_{i,j}$ in local coordinates | $\mathbf{p}' \in \mathbb{R}^3$ |
| Geometric center of $P_{i,j}$ in global coordinates | $\mathbf{c}_{i,j} \in \mathbb{R}^3$ |
| Contact between part $P_{i,j}$ and $P_{i,k}$ | $C_{i,j,k}$ |
| Neighborhood graph for shape $S_i$ | $\mathcal{G}_i$ |
| Edge in $\mathcal{G}_i$ corresponding to $C_{i,j,k}$ | $e_{i,j,k} \in \mathcal{E}$ |

we can do a principal component analysis (PCA) for each part, and thus estimate a local co-ordinate axes for each part of the object.

The individual points of $\bar{P}_{i,j}$ are placed on the surface using the Poisson-disk sampling algorithm, meaning each sampled point on the surface must be *disk-free*, i.e. at least a minimum distance, $r$, from any other previous sample point. This gives a random set of points on the surface such that the points are tightly packed together, but, are no closer than a specified minimum distance. This gives us point samples from the surface such that their distance is roughly equal, and exhibits blue noise characteristics in its spectrum.

### 2.2.3 PCA Analysis and Local Co-ordinate System for a Part

Now, a principal component analysis (PCA) of $\bar{P}_{i,j}$ is performed, which provides a transformation $\mathsf{T}_{i,j}$ from the global into the local coordinate system of the part. A point $\mathbf{p}'$ in the local coordinate system is given by a transformation $\mathsf{T}_{i,j}$, which combines a translation, a rotation and a scaling: $\mathbf{p}' = \mathsf{T}_{i,j}\mathbf{p} = \mathsf{S}_{i,j}\mathsf{R}_{i,j}(\mathbf{p} - \mathbf{c}_{i,j})$. The geometric center of the part's point cloud in the world coordinate system defines the center $\mathbf{c}_{i,j}$ of the local coordinate system. The local $3 \times 3$ rotation matrix $\mathsf{R}_{i,j}$ is given by the three PCA basis vectors and defines the local rotation axes. The diagonal $3 \times 3$ matrix $\mathsf{S}_{i,j} = \mathrm{diag}(1/s_{\mathrm{x}}, 1/s_{\mathrm{y}}, 1/s_{\mathrm{z}})$ describes the local non-uniform inverse scaling using the three singular-values $\mathbf{s}_{i,j} = (s_{\mathrm{x}}, s_{\mathrm{y}}, s_{\mathrm{z}})^{\top}$, which can be seen in Figure 2.6. Table 2.2 lists all symbols used throughout this section.

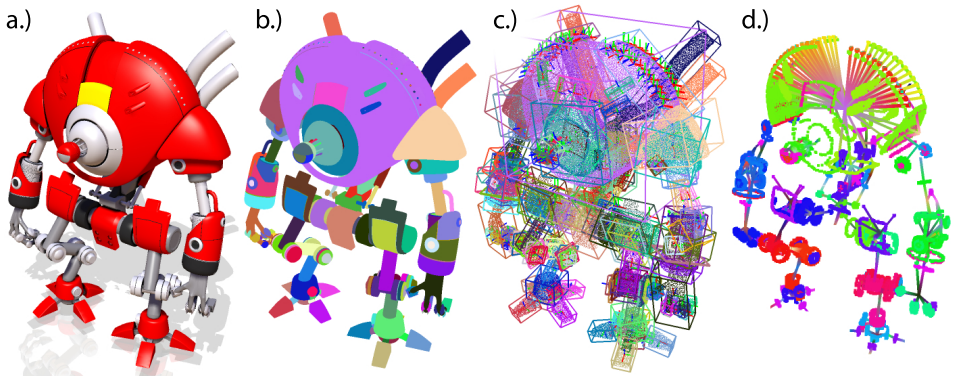



Figure 2.6: *a.)* Shape, *b.)* Segmentation, *c.)* PCA, *d.)* Contacts.

### 2.2.4 Contact Analysis

During contact analysis all intersections of all parts for a shape are found. For each part $P_{i,j}$ of shape $S_i$ it is evaluated, if it is in contact with another part $P_{i,k}$. We call the subset of points in the point cloud $\bar{P}_{i,j}$ for which a point with a distance of less than 0.1 % of the bounding box diameter exists in $\bar{P}_{i,k}$ the *contact* $C_{i,j,k}$ of part $j$ and $k$. In practice, an axis-aligned bounding box tree [van den Bergen 1998] on all points of shape $S_i$ is used to compute the set of contact points efficiently. A graph $\mathcal{G}_i = (\mathcal{V}, \mathcal{E})$ which encodes the adjacency structure of the 3D object can now build using the set of contacts. As shown in Figure 2.7, the set of nodes $\mathcal{V}$ is defined by all parts $P_{i,j} \in \bigcup_{j=1}^{j=n_i^{\mathrm{p}}} P_{i,j}$ in the object $S_i$. If there is a contact $C_{i,j,k}$ between parts $P_{i,j}$ and another part $P_{i,k}$ of the shape $S_i$, we add an edge $e_{i,j,k}$ to the set of edges $\mathcal{E}$. All this information is computed in a pre-processing step and serialized to disk.

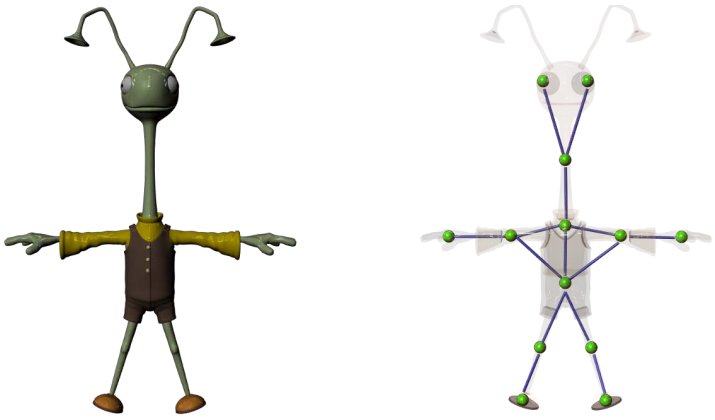


Figure 2.7: The contact graph is created by adding an edge between parts which share a contact.

In summary, the above steps produce a segmentation of each shape into parts, each with their local coordinate system, as well as a list of contacts between the parts of a shape.

# 3

# Previous Work

In this section we review the related work in the areas of data-driven video editing, 3D modeling and material assignment. Our discussion on previous work on data-driven content creation techniques starts with Section 3.1 where we review various approaches employed for statistical analysis on collections of 3D meshes. Thereafter, we look at related work in specific fields where we applied our data-driven models. In Section 3.2 we summarize existing literature in the field of video manipulation and monocular video tracking. We then look at novel methods for 3D modeling in Section 3.3 and finally at material and texture assignment techniques in Section 3.4.

## 3.1 Statistical Analysis on Collections of 3D Meshes

We treat *i*) collections of single-part objects and *ii*) collections of multi-part objects separately in Section 3.1.1 and Section 3.1.2.

Single-part objects are 3D models without any segmentation information. Collections of such single-part objects are usually homogeneous, and a complete one-to-one correspondence across the complete mesh collection can be established.

Multi-part objects are models which are implicitly composed out of many discrete parts, and this information is readily available (or can easily extracted by pre-processing the models). Such collections tend to be heterogeneous, and establishing complete correspondences across such unstructured collections is seldom feasible in practice.

### 3.1.1 Collections of Single-Part Meshes

A collection of shapes with assigned mesh correspondences (e.g. corresponding mesh vertices) inherently encodes a deformation model for the collection. Such

shapes can be seen as points in a common high dimensional vector space. For such shapes, their principal modes of variations can be directly extracted using statistical tools such as PCA. Blanz et al. in their highly influential work [Blanz and Vetter 1999] first explored this idea in the context of 3D face models. In our approach we build a morphable model of human shape and pose similar to [Allen, Curless and Popović 2003; Seo and Magnenat-Thalmann 2004; Anguelov et al. 2005; Allen et al. 2006; Hasler et al. 2009]. This model has been learned from a publicly available database of human body scans in different poses that is kindly provided by Hasler et al. [Hasler et al. 2009]. Our body model is a variant of the SCAPE model by Anguelov et al. [Anguelov et al. 2005] that describes body shape variations with a linear PCA model. Since SCAPE's shape PCA dimensions do not correspond to semantically meaningful dimensions, we remap the body parameters to semantically meaningful attributes through a linear regression similar to Allen et al. [Allen, Curless and Popović 2003].

Sumner et al. [Sumner et al. 2005] learn the deformation space for a mesh by using a set of example meshes to indicate the class of meaningful deformation. Each example is represented with a feature vector of deformation gradients that capture the transformations relative to a reference pose. They then allow for direct mesh deformation, by searching the closest pose match in a nonlinear span of the example feature vectors. Kokkinos and Yuille [Kokkinos and Yuille 2007] use an active appearance model to learn object deformation models using objects with manually annotated landmark correspondences. In medical imaging, Kim et al. [Kim et al. 2012] use a statistical correlation model learned between image appearances and deformation field for robust registration of Magnetic Resonace(MR) brain scans. Similarly, Kilian et al. [Kilian, Mitra and Pottmann 2007] create a shape space that allow the computation of geodesic paths between model pairs with given correspondence for the restricted class of isometric deformations. Mitra et al. [Mitra, Guibas and Pauly 2007] achieve object deformation by indirectly mapping transformation domain manipulations to corresponding shapes in the object space.

### 3.1.2 Collections of Multi-Part Meshes

The process of creating a new shape that interpolates two given shapes is called blending, or morphing. Morphing involves solving a correspondence problem and requires a blending operator. Due to the combinatorial complexity, finding correspondences is a difficult problem both between images [Wolberg 1998] and surfaces [Beier and Neely 1992; Lee et al. 1999]. Blending becomes easier when choosing a suitable representation, such as distance fields [Cohen-Or, Solomovic and Levin 1998] and achieves more natural results when maintaining as-rigid-as-possible deformations [Alexa, Cohen-Or and Levin 2000].

Interpolating between multiple given models is even more challenging. The problem is simplified when a parametric model is available, as is the case for human faces

and bodies [Blanz and Vetter 1999; Allen, Curless and Popović 2003](as discussed in Section 2.1, but currently no general solution exists to construct such parametric spaces given general models composed of many meaningful parts.

Another approach to shape synthesis is to generate instances based on statistical models. This is a popular approach for texture synthesis, where example instances are decomposed into multiple texture elements and then recombined into new instances [Efros and Leung 2002]. This approach has been extended to synthesize large models given a smaller one [Merrell 2007]; however, when synthesized instances must have a complex hierarchical structure, such as symmetry and preservation of physical constraints, statistical models based on local similarity are less effective, and some modeling of the global structure is required.

To circumvent the difficulty of establishing correspondences across heterogeneous collections of shapes, several alternate strategies have been proposed. Text keywords and shape part proxies have been employed by [Funkhouser et al. 2004; Chaudhuri and Koltun 2010] for querying such collections for similar shapes. Alternatively, suitable global descriptors can be extracted from the models (e.g., shape distributions [Osada et al. 2002], spherical harmonics [Kazhdan, Funkhouser and Rusinkiewicz 2003], spherical wavelets [Laga, Takahashi and Nakajima 2006], heat kernels [Ovsjanikov et al. 2009]. Based on the properties of the descriptors (e.g., shape, pose or rotation invariance), shapes are embedded in a consistent descriptor space without requiring explicit object level correspondences. Recently, part-based correspondence has been explored as a method for studying variations within such model clusters. For example, [Golovinskiy and Funkhouser 2009] present an algorithm for simultaneously segmenting while establishing part correspondences across a set of models based on clustering on a graph of potential corresponding model polygons. [Kalogerakis, Hertzmann and Singh 2010] introduce a data-driven approach for learning a consistent segmentation and labeling of model parts using a range of geometric and contextual features. [Ovsjanikov et al. 2011] demonstrate that models variations can be learned and explored from model collections by establishing a mapping between descriptor space and object space without computing direct correspondences at the level of features or parts.

Procedural modeling systems have been used to describe the global hierarchical structure in shapes [Ebert 2003], but it is difficult to automatically derive a grammar for a given geometric model, despite promising recent steps [Bokeloh, Wand and Seidel 2010].

## 3.2 Data-driven Video Editing

In our work we can capitalize on previous research from a variety of areas. Exemplary work from video retouching, marker-less pose and motion estimation and extending existing training sets for articulated human detection and pose estimation

is briefly reviewed in the following.

**Video Retouching** Several commercial-grade image manipulation tools exist[1] that enable a variety of basic retouching operations, such as segmentation, local shape editing, or compositing. The research community also worked on object-based manipulation approaches that broaden the scope of the above basic tools, e.g., [Barrett and Cheney 2002]. Unfortunately, more advanced image edits are very cumbersome with the aforementioned approaches. A solution is offered by semantically-guided image operations, in which some form of scene model represents and constrains the space of permitted edits, such as a face model for automatic face beautification [Leyvand et al. 2008], or a body model for altering body attributes in photographs [Zhou et al. 2010].

Applying similarly complex edits to entire video streams is still a major challenge. The *Proscenium* system by Bennett et al. [Bennett and McMillan 2003] allows the user to shear and warp the video volumes, for instance to stabilize the camera or remove certain objects. [Liu et al. 2005] describe an algorithm for amplification of apparent motions in image sequences captured by a static camera. Wang et al. [Wang et al. 2006] present the cartoon animation filter that can alter motions in existing video footage such that it appears more exaggerated or animated. Spatio-temporal gradient domain editing enables several advanced video effects, such as re-compositing or face replacement, at least if the faces remain static [Wang et al. 2007]. Spatio-temporal segmentation of certain foreground objects in video streams also paves the trail for some more advanced edits, such as repositioning of the object in the field of view [Wang et al. 2005; Li, Sun and Shum 2005]. However, none of these methods enables easy complete reshaping of human actors in a way similar to the algorithm presented in this thesis.

Our system has parallels to video retargeting algorithms that allow, for instance, to resize video while keeping the proportions of visually salient scene elements intact. Two representative video retargeting works are [Krähenbühl et al. 2009; Rubinstein, Shamir and Avidan 2008]. However, complex plausible reshaping of humans in video is not feasible with these approaches.

Our approach employs a morphable model of human shape and pose to guide the reshaping of the actor in the video sequence. Conceptually related is the work by Scholz et al. who use a model of moving garment to replace clothing textures in monocular video [Scholz et al. 2009]. Vlasic et al. [Vlasic et al. 2005] employ a morphable 3D face model to transfer facial expressions between two video sequences, where each one is showing a different individual. Finally, [Scholz and Magnor 2006] describe an algorithm to segment video objects and modify their motion within certain bounds by editing some key-frames. The algorithm by Hornung et al. [Hornung, Dekkers and Kobbelt 2007] solves a problem that

[1] e.g. Adobe Photoshop™, GIMP, etc.

is kind of opposite to what we aim for. They describe a semi-automatic method for animation of still images that is based on image warping under the control of projected 3D motion capture data. None of the aforementioned approaches could perform semantically plausible reshaping of actors in video footage in a similar manner as our approach.

**Marker-less Pose and Motion Estimation** Monocular pose estimation from images and video streams is a highly challenging and fundamentally ill-posed problem. A few automatic approaches exist that attack the problem in the monocular case [Agarwal and Triggs 2006]. However, they often deliver very crude pose estimates and manual user guidance is required to obtain better quality results, e.g., [Davis et al. 2003; Parameswaran and Chellappa 2004; Hornung, Dekkers and Kobbelt 2007]. Recently, Wei and Chai [Wei and Chai 2010] presented an approach for interactive 3D pose estimation from monocular video. Similar, as with our approach in the monocular video case, manual intervention in a few keyframes is required.

In our research, we apply a variant of the marker-less pose estimation algorithm by [Gall et al. 2009] for pose inference in video. Our approach is suitable for both monocular and multi-view pose inference. A variety of marker-less motion estimation algorithms for single and multi-view video have been proposed in the literature, see [Poppe 2007] for an extensive review. Many of them use rather crude body models comprising skeletons and simple shape proxies that would not be detailed enough for our purpose. At the other end of the spectrum, there are performance capture algorithms that reconstruct detailed models of dynamic scene geometry from multi-view video [de Aguiar et al. 2008; Vlasic et al. 2008]. However, they solely succeed on multi-view data, often require a full-body scan of the tracked individual as input, and do not provide a plausible parameter space for shape manipulation.

Therefore, our algorithm is based on a morphable human body model as described in the previous paragraph. Only a few other have employed such a model for full-body pose capture. Balan et al. [Balan et al. 2007] track the pose and shape parameters of the SCAPE model from multi-view video footage. So far, monocular pose inference with morphable models has merely been shown for single images, [Guan et al. 2009; Hasler et al. 2010; Zhou et al. 2010; Sigal, Balan and Black 2007; Rosales and Sclaroff 2006], where manual intervention by the user user is often an integral part of the pipeline. In contrast, in our video retouching algorithm we estimate time-varying body shape and pose parameters from both single and multi-view footage, with only a small amount of user intervention needed in the monocular video case.

**Extending Existing Training Sets for Articulated Human Detection and Pose Estimation** Training state-of-the-art models for people detection of strongly artic-

ulated people in images or video requires representative training sets. Collecting and annotating such data sets is tedious and many images are required for good performance [Johnson and Everingham 2011]. Here, we follow the appealing route to generate training data based on computer graphics methods. Automatically generated data has been used in computer vision in the past. However, its application has been mostly limited to cases where realistic appearance is not required, such as silhouette-based methods for human pose estimation [Agarwal and Triggs 2006] or depth images [Shotton et al. 2011]. While training people detectors from rendered images has been proposed [Okada and Soatto 2008; Marin et al. 2010; Shakhnarovich, Viola and Darrell 2003], such training data often lacks the necessary realism for good performance. An alternative is to apply transformations to real images preserving their realism. E.g. [Enzweiler and Gavrila 2008] augments the training set by applying a morphable 2D model to images. Here we follow a similar idea, however in our case we use a generative 3D human shape model and motion capture data to generate possible deformations of 2D data making our deformation model more realistic and versatile. [Pishchulin et. al. 2011a], [Zhou et al. 2010] are probably closest to our work. In our own prior work [Pishchulin et. al. 2011a] we require an expensive data acquisition step limiting the number of subjects in the experiments to a handful of people. Both methods are limited to shape deformations *only*. On the contrary in this work we are able to generate new training examples from existing 2D images while still allowing for a wide range of shape *and* pose variations. We show that controlling pose variations of generated training samples is essential when training detection models of highly articulated people.

## 3.3 Data-driven 3D Modeling

There is a significant amount of prior work that supports modeling with the help of a database of existing models [Funkhouser et al. 2004; Chaudhuri and Koltun 2010; Fisher and Hanrahan 2010; Xu et al. 2010]. While some of these techniques propose or use specific ways of searching the database for good candidate models (or suggestions), these methods typically rely on example-based search, which, as discussed earlier, is not ideal for exploring unfamiliar collections of shapes.

Modeling by Example [Funkhouser et al. 2004] is a modeling system that can be used to generate a shape by manually cutting and gluing parts from existing shapes. Chaudhuri et al. [Chaudhuri et al. 2011] propose an example-based modeling system that expedites the modeling process by presenting relevant components to the user. Our goal is to further drastically simplify the modeling interface by synthesizing complete models automatically and efficiently, allowing the user to explore a whole family of new models with no direct manipulation of geometry. The *Shuffler* system [Kraevoy, Julius and Sheffer 2007] enables users to replace selected parts in compatible models. Section 5.4 discusses similarities and differences to our system. To efficiently explore large databases of 3D shapes, Ovsjanikov et al.

[Ovsjanikov et al. 2011] propose a navigation interface that allows users to find the desired shape in the database by interacting with a deformable shape template.

Higher-level shape analysis has received much interest with applications that include deformation [Gal et al. 2009; Zheng et al. 2011], abstractions [Mehra et al. 2009], automated layout [Li et al. 2008], or upright orientation [Fu et al. 2008]. A crucial component here is the segmentation of meshes [Sharf et al. 2006; Golovinskiy and Funkhouser 2009; Kalogerakis, Hertzmann and Singh 2010] and the detection of symmetry [Mitra, Guibas and Pauly 2006; Pauly et al. 2008]. Symmetries can also be organized hierarchically [Wang et al. 2011]; joints between shape parts can be automatically extracted [Xu et al. 2009]; and mechanical assemblies can be animated given only raw shapes as input [Mitra et al. 2010].

## 3.4 Data Driven Material Assignment

Exploiting the relations of materials and shapes so far received only limited attention in the computer vision community, and even less interest from computer graphics. For editing and assigning material, different material design interfaces are employed. In most commercial 3D modeling tools it is still common to directly modify the parameters of the analytic reflectance model such as Phong [Phong 1975]. Ngan et al. [Ngan, Durand and Matusik 2006] propose an interface for BRDF selection that displays material variations with several preview images. There are also special solutions [Kautz, Boulos and Durand 2007; Pellacini and Lawrence 2007] to edit spatially-varying material representations. Kerr and Pellacini [Kerr and Pellacini 2010] have performed a user-study to evaluate material design interfaces with either physical sliders, perceptual sliders, or preview image navigation. We now outline some recent advances in texture transfer and data-driven content creation.

**Texture Transfer** Closely related to our approach are texture transfer methods [Mertens et al. 2006; Lu et al. 2007]. These approaches model the statistical relationship between local geometric properties, such as curvature and local statistics of reflectance. Using this relation, a texture synthesis on a new object produces shape-dependent textures that capture, e. g., weathering. Our approach is different, as it is neither considering the statistics of local shape descriptors nor the statistics of reflectance. Instead, we work on high-level structure, such as spatial arrangement, shape, and material similarities to capture the global organization instead of local statistics. Chajdas et al. [Chajdas, Lefebvre and Stamminger 2010] propose a system that assists a user to assign textures in large virtual environments. A user-provided texture assignment is automatically propagated to similar surfaces in the environment. Textures can also be re-targeted to different surface sizes [Lefebvre, Hornus and Lasram 2010]. In contrast, our approach focuses on material properties that are extracted from a database, such as diffuse color, specularity, glossiness, and

transparency. We assume that a material is constant for a part of an object and ignore spatially varying properties typically stored in textures.

**Data-driven Content Creation** Supporting artists or casual users to effectively create content has recently received much interest. Our approach is inspired by other data-driven 3D content creation tools, e. g., for 3D modeling [Chaudhuri et al. 2011], hand-drawings [Lee, Zitnick and Cohen 2011], furniture arrangement [Yu et al. 2011], image color themes [Wang et al. 2010], or segmentation and labeling of objects [Kalogerakis, Hertzmann and Singh 2010]. Fisheret al. [Fisher and Hanrahan 2010; Fisher, Savva and Hanrahan 2011] use a database of objects to search for a suitable object that fits into a given spatial context. They employ the Visual Memex Model from Malisiewicz and Efros [Malisiewicz and Efros 2009] that stores associations between entities instead of categorizing them. The Material Memex model presented in this thesis also adopts this methodology but applies it in the application domain of material assignment. Furthermore, because our approach estimates the materials of several parts of an object simultaneously, a more complex probabilistic framework must be employed.

# 4

# Data-driven Video Editing

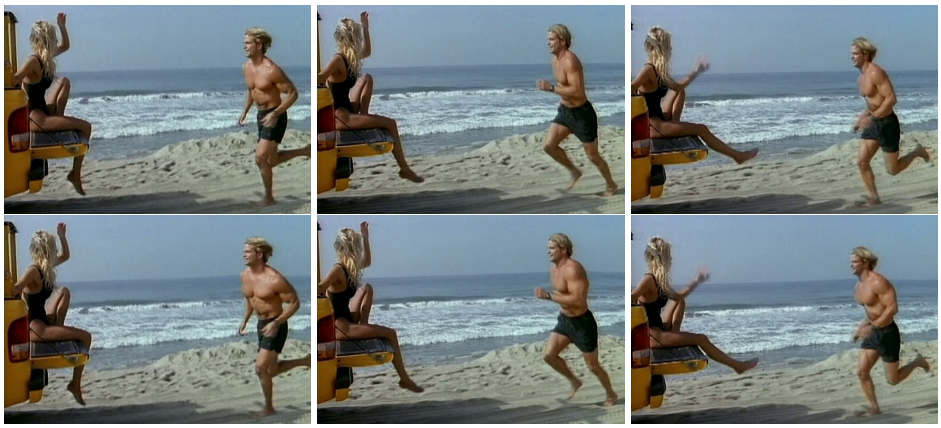


**Figure 4.1:** In this sequence from the TV series Baywatch, we modified the original appearance of the actor (top row) such that he appears more muscular (bottom row). The edit was performed with our system by simply increasing the value on the muscularity control slider.

## 4.1 Introduction

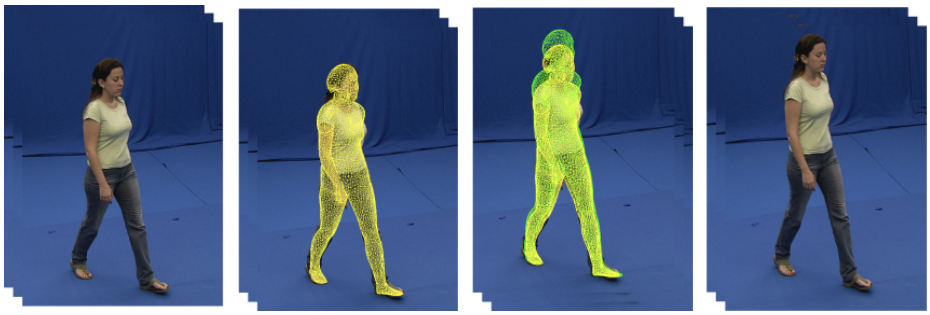
Digital retouching of photographs is an essential operation in commercial photography for advertisements or magazines, but is also increasingly popular among hobby photographers. Typical retouching operations aim for visual perfection, for instance by removing scars or birth marks, adjusting lighting, changing scene backgrounds, or adjusting body proportions. Unfortunately, even commercial-grade image editing tools often only provide very basic manipulation functionality. Therefore, many advanced retouching operations, such as changing the appearance or proportions of the body, often require hours of manual work. To facilitate such advanced editing operations, researchers developed semantically-based retouching tools that employ

parametric models of faces and human bodies in order to perform complicated edits more easily. Examples are algorithms to increase the attractiveness of a face [Leyvand et al. 2008], or to semi-automatically change the shape of a person in a photograph [Zhou et al. 2010].

While such semantically-based retouching of photographs is already very challenging, performing similar edits on video streams has almost been impossible up to now. Existing commercial video editing tools (Section 3.2) only provide comparatively basic manipulation functions, such as video object segmentation or video retargeting, and already these operations are computationally very demanding. Only a few object-based video manipulation approaches go slightly beyond these limits, for instance by allowing facial expression change [Vlasic et al. 2005], modification of clothing texture [Scholz and Magnor 2006], or by enabling simple motion edits of video objects [Scholz et al. 2009]. The possibility to easily manipulate attributes of human body shape, such as *weight*, *height* or *muscularity*, would have many immediate applications in movie and video post-production. Unfortunately, even with the most advanced object-based video manipulation tools, such retouching would take even skilled video professionals several hours of work. The primary challenge is that body shape manipulation, even in a single video frame, has to be performed in a *holistic* way. Since the appearance of the entire body is strongly correlated, body reshaping solely based on local operations is very hard. As an additional difficulty, body reshaping in video has to be done in a spatio-temporally coherent manner.

We therefore propose in one of the first systems in the literature to easily perform holistic manipulation of body attributes of human actors in video. Our algorithm is based on a 3D morphable model of human shape and pose that has been learned from full body laser scans of real individuals. This model comprises a skeleton and a surface mesh. Pose variation of the model is described via a standard surface skinning approach. The variation of the body shape across age, gender and personal constitution is modeled in a low-dimensional principal-component-analysis (PCA) parameter space. A regression scheme enables us to map the PCA parameters of human shape onto semantically meaningful scalar attributes that can be modified by the user, such as: *height*, *waist girth*, *breast girth*, *muscularity*, etc. In a first step, a marker-less motion estimation approach spatio-temporally optimizes both the pose and the shape parameters of the model to fit the actor in each video frame. In difficult poses, the user can support the algorithm with manual constraint placement. Once the 3D model is tracked, the user can interactively modify its shape attributes. By means of an image-based warping approach, the modified shape parameters of the model are applied to the actor in each video frame in a spatio-temporally coherent fashion.

We illustrate the usefulness of our approach on *single-view* and *multi-view video* sequences. For instance, we can quickly and easily alter the appearance of actors in existing movie and video footage. Further on, we can alter the physical attributes of



(a) Input video (b) Tracking (c) Reshaping (d) Output video

**Figure 4.2:** The two central processing steps of our system are **tracking** and **reshaping** of a morphable 3D human model.

actors captured in a controlled multi-view video studio. This allows us to carefully plan desired camera viewpoints for proper compositing with a virtual background, while giving us the ability to arbitrarily re-touch the shape of the actor during post-processing. We also confirmed the high visual fidelity of our results in a user study.

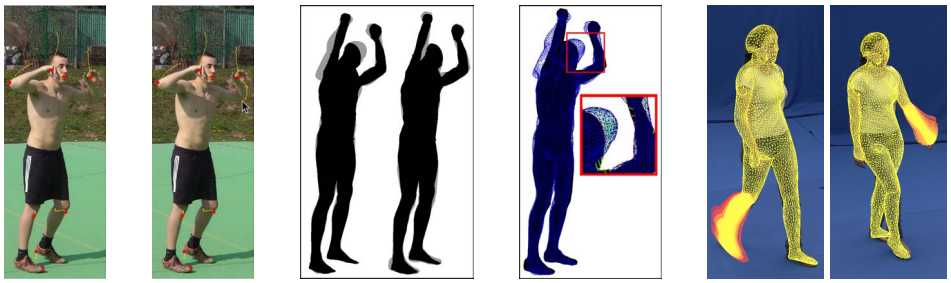
## 4.2 Overview

Our system takes as input a *single-view* or *multi-view* video sequence with footage of a human actor to be spatio-temporally reshaped (Figure 4.2). There is no specific requirement on the type of scene, type of camera, or appearance of the background.

(a) (b)

**Figure 4.3:** Morphable body model - (a) Samples of the pose and shape parameter space that is spanned by the model. (b) The average human shape with the embedded kinematic skeleton.

As a first step, the silhouette of the actor in the video footage is segmented using off-the-shelf video processing tools. The second step in the pipeline is marker-less model fitting. There, both the shape and the pose parameters of the 3D model are optimized such that it re-projects optimally into the silhouette of the actor in each video frame (Section 4.3). Once the model is tracked, the shape parameters of the actor can be modified by simply tweaking a set of sliders corresponding to individual semantic shape attributes. Since the original PCA parameter dimensions of the morphable shape model do not directly correspond to plausible shape attributes, we learn a mapping from intuitive attributes, such as muscularity or weight, to the underlying PCA space (Section 4.4.1). Now reshaping can be performed by adjusting plausible parameter values. Once the target set of shape attributes has been decided on, they are applied to the actor in all frames of the video input by performing image-based warping under the influence of constraints that are derived from the re-projected modified body model (Section 4.4.2).



(a) (b) (c) (d) (e)

**Figure 4.4: (a)-(d) Components of the pose error function:** (a) KLT features and their trajectories (yellow) over several frames; (b) in the monocular video case, additional feature point tracks can be manually generated or broken trajectories can be linked; (c) silhouette error term used during global optimization; a sum of image silhouette pixels not covered by the model, and vice versa (erroneous pixels in dark grey), (d) silhouette error term used during local optimization - corresponding points between image and model silhouettes and their distances are shown; **(e) Global pose optimization:** sampled particles (model pose hypotheses) are overlaid for the leg and the arm.

## 4.3 Tracking with a Statistical Model of Pose and Shape

In the following, we review the details of the 3D human shape model, and explain how it is used for tracking the actor in a video.

### 4.3.1 3D Morphable Body Model

As also outlined in Section 2.1 we employ a variant of the SCAPE model [Anguelov et al. 2005] to represent the pose and the body proportions of an actor in 3D. We learned this model from a publicly available database of 550 registered body scans of over 100

people (roughly 50% male subjects, and 50% female subjects, aged 17 to 61) in different poses (Figure 4.3(a)). The motion of the model is represented via a kinematic skeleton comprising of 15 joints. The surface of the model consists of a triangle mesh with roughly 6500 3D vertices $\mathbf{v_i}$. As opposed to the original SCAPE model, we do not learn per-triangle transformation matrices to represent subject-specific models of pose-dependent surface deformation. In our application, this level of detail is not required to obtain realistic reshaping results. Further on, the omission of this per-triangle model component prevents us form having to solve a large linear system to reconstruct the model surface, every time the model parameters have changed. This, in turn, makes pose estimation orders of magnitude faster. Instead of per-triangle transformations, we use a normal skinning approach for modeling pose-dependent surface adaptation. To this end, the skeleton has been rigged into the average shape human shape model by a professional animation artist (Figure 4.3(b)).

Similar to the original SCAPE model, we represent shape variation across individuals via principal component analysis (PCA). We employ the first 20 PCA components which capture 97% of the body shape variation. In total, our model thus has $N = 28$ pose parameters $\Phi = (\phi_1, \ldots, \phi_N)$ and $M = 20$ parameters $\Lambda = (\lambda_1, \ldots, \lambda_M)$ to represent the body shape variation.

### 4.3.2 Marker-less Tracking

We use a marker-less motion capture approach to fit the pose and shape of the body model to a human actor in each frame of a single-view or multi-view video sequence. In case the input is an arbitrary monocular video sequence, we make the simplifying assumption that the recording camera is faithfully modeled by a scaled orthographic projection. In the multi-view video case we expect fully-calibrated frame-synchronized cameras, which is a reasonable assumption to make as most of these sequences are captured under controlled studio conditions.

Henceforth, we denote a video frame at time stamp $t$ seen from camera $c$ ($c = 1, \ldots, C$) with $I_{t,c}$. $\Phi_t$ are the pose parameters and $\Lambda_t$ are the body shape parameters at time stamp $t$. Before tracking commences, the person is segmented from the background in each video frame, yielding a foreground silhouette. To serve this purpose, we rely on standard video processing tools[1] if chroma-keying is not possible, but note that alternative video object segmentation approaches, such as [Wang et al. 2005; Li, Sun and Shum 2005], would be equally applicable.

Our motion capture scheme infers pose and shape parameters by minimizing an image-based error function $E(\Phi, \Lambda, t)$ that, at each time step of video $t$, penalizes misalignment between the 3D body model and its projection into each frame:

$$E(\Phi_t, \Lambda_t) = \sum_{c=1}^{C} E_s(\Phi, \Lambda_t, I_{t,c}) + E_f(\Phi_t, \Lambda_t, I_{t,c}) \; . \tag{4.1}$$

[1] Mocha™, Adobe AfterEffects™

Let the binary silhouette for the image $I_{t,c}$ be $S^{gt}_{t,c}$ and let $S^{proj}_{t,c}$ be the silhouette created by projection of the 3D mesh into camera $c$. Let the set of $K$ 3D points on the model surface be $\mathbf{v_i}$ and their corresponding locations in the video frame at time $t$ in camera $c$ be $\mathbf{u}_{i,c}$. Let the corresponding KLT tracked point be $\mathbf{u}^{klt}_{i,c}$

The first component $E_s = \sum_c^C \|S^{gt}_{t,c} - S^{proj}_{t,c}\|_2^2$ measures the misalignment of the silhouette boundary of the re-projected model with the silhouette boundary of the segmented person. The second component $E_f = \sum_c^C \sum_i^K \|\mathbf{u}_{i,c} - \mathbf{u}^{klt}_{i,c}\|_2^2$ measures the sum of distances in the image plane between feature points of the person tracked over time, and the re-projected 3D vertex locations of the model that - in the previous frame of video - corresponded to the respective feature point. Feature trajectories are computed for the entire set of video frames before tracking commences (Figure 4.4(a)). To this end, an automatic Kanade-Lucas-Tomasi (KLT) feature point detector and tracker is applied to each video frame. Automatic feature detection alone is often not sufficient, in particular if the input is a monocular video: Trajectories easily break due to self-occlusion, or feature points may not have been automatically found for body parts that are important but contain only moderate amounts of texture. We therefore provide an interface in which the user can explicitly mark additional image points to be tracked, and in which broken trajectories can be linked (Figure 4.4(b)).

Pose inference at each time step $t$ of a video is initialized with the pose parameters $\Phi_{t-1}$ and shape parameters $\Lambda_{t-1}$ determined in the preceding time step. For finding $\Phi_t$ and $\Lambda_t$ we adapt the combined local and global pose optimization scheme by [Gall et al. 2009].

Given $\mathbf{v_i}$, a fast local optimization is first performed to determine the pose parameters of each body part. During local optimization, $E_s$ in Eq. (4.1) is computed by assigning a set of points on the model silhouette to the corresponding closest points on the image silhouette, and summing up the 2D distances (Figure 4.4(c)).

Each 2D point $\mathbf{u}_{i,c}$ defines a projection ray that can be represented as a Plücker line $L_{i,c} = (n_{i,c}, m_{i,c})$ [Stolfi 1991]. The error of pair $(\mathcal{T}(\Phi_t, \Lambda_t)\mathbf{v}_i, \mathbf{u}_{i,c})$ is given by the norm of the perpendicular vector between the line $L_i$ and the 3D point $\mathbf{v}_i$ from the body models standard pose, transformed by transformation $\mathcal{T}(\Phi_t, \Lambda_t)$ that concatenates the pose, shape, and skinning transforms. Finding the nearest local pose and shape optimum of Eq. (4.1) therefore corresponds to solving

$$\underset{(\Phi_t, \Lambda_t)}{\operatorname{argmin}} \sum_c^C \sum_i^K w_i \|\Pi(\mathcal{T}(\Phi_t, \Lambda_t)\mathbf{v}_{i,c}) \times n_{i,c} - m_{i,c}\|_2^2 \tag{4.2}$$

which is linearized using Taylor approximation and solved iteratively. $\Pi$ is the projection from homogeneous to non-homogeneous coordinates.

Local pose optimization is extremely fast but may in some cases get stuck in incorrect local minima. Such pose errors could be prevented by running a full global pose optimization. However, global pose inference is prohibitively slow

when performed on the entire pose and shape space. We therefore perform global pose optimization only for those sub-chains of the kinematic model, which are incorrectly fitted. Errors in the local optimization result manifest through a limb-specific fitting error $E(\Phi_t, \Lambda_t)$ that lies above a threshold [Gall et al. 2009]. For global optimization, we utilize a particle filter. Figure 4.4(d) overlays the sampled particles (pose hypotheses) for the leg and the arm.

In practice, we solve for pose and shape parameters in a hierarchical way. First, we solve for both shape and pose using only a subset of key frames of the video in which the actor shows a sufficient range pose and shape deformation. It turned out that in all our test sequences the first 20 frames form a suitable subset of frames. In this first optimization stage, we solely perform global pose and shape optimization and no local optimization. Thereafter, we keep the shape parameters fixed, and subsequently solve for the pose in *all* frame using the combined local and global optimization scheme.

We employ the same tracking framework for both multi-view ($C > 1$) and single view video sequences ($C = 1$). While multi-view data can be tracked fully-automatically, single view data may need more frequent manual intervention. In all our monocular test sequences, though, only a few minutes of manual user interaction were needed. Please note that monocular pose tracking is ill-posed, and therefore we cannot guarantee that the reconstructed model pose and shape are correct in a metric sense. However, in our retouching application such 3D pose errors can be tolerated as long as the re-projected model consistently overlaps with the person in all video frames. Also, for our purpose it is not essential that the re-projected model aligns exactly with the contours of the actor. The image-based warping deformation described in the following also succeeds in the presence of small misalignments.

## 4.4 Reshaping Interface

Once tracking information for shape and pose has been obtained, the body shape of the actor can be changed with our interactive reshaping interface (see Figure 4.5).

### 4.4.1 Deformation of Human Shape

The PCA shape space parameters $\Lambda$ do not correspond to semantically meaningful dimensions of human constitution. The modification of a single PCA parameter $\lambda_k$ will simultaneously modify a combination of shape aspects that we find intuitively plausible, such as *weight* or *strength of muscles*. We therefore remap the PCA parameters onto meaningful scalar dimensions. Fortunately, the scan database from which we learn the PCA model contains for each test subject a set of semantically meaningful attributes, including: *height*, *weight*, *breast girth*, *waist girth*, *hips girth*, *leg length*, and *muscularity*. All attributes are given in their respective measurement

units, as shown in Figure 4.5.

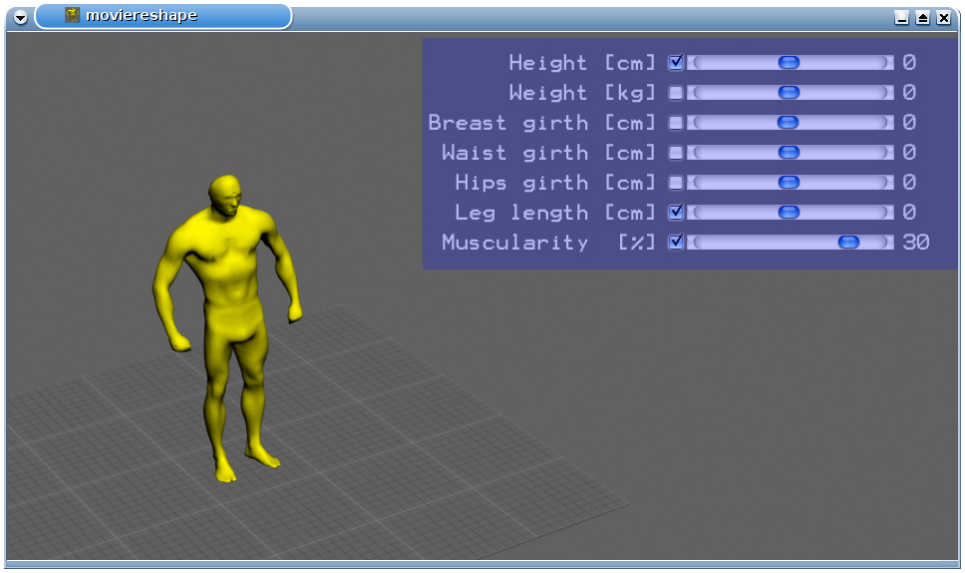



**Figure 4.5:** The reshaping interface allows the user to modify semantic shape attributes of a person.

Similar to [Allen, Curless and Popović 2003] we project the $Q = 7$ semantic dimensions onto the $M$ PCA space dimensions by constructing a linear mapping $\mathbf{S} \in \mathcal{M}((M-1) \times (Q+1))$ between these two spaces:

$$\mathbf{S}\,[f_1 \ \ldots \ f_Q \ 1]^T = \Lambda\,, \tag{4.3}$$

where $f_i$ are the semantic attribute values of an individual, and $\Lambda$ are the corresponding PCA coefficients. This mapping enables us to specify offset values for each semantic attribute $\Delta\mathbf{f} = [\Delta f_1 \ \ldots \ \Delta f_Q \ 0]^T$. By this means we can prescribe by how much each attribute value of a specific person we tracked should be altered. For instance, one can specify that the weight of the person shall increase by a certain amount of kilograms. The offset feature values translate into offset PCA parameters $\Delta\Lambda = \mathbf{S}\Delta\mathbf{f}$ that must be added to the original PCA coefficients of the person to complete the edit.

Please note that certain semantic attributes are implicitly correlated to each other. For instance, increasing a woman's height may also lead to a gradual gender change since men are typically taller than women. In an editing scenario, such side-effects may be undesirable, even if they would be considered as generally plausible. In the end, it is a question of personal taste which correlations should be allowed to manifest and which ones should be explicitly suppressed. We give the user control over this decision and give him the possibility to explicitly fix or let free certain attribute dimensions when performing an edit. To start with, for any attribute value our reshaping interface provides reasonable suggestions of what parameters to fix when modifying certain attributes individually. For instance, one suggestion is that when editing the height, the waist girth should be preserved.

### 4.4.2 Consistent Video Deformation

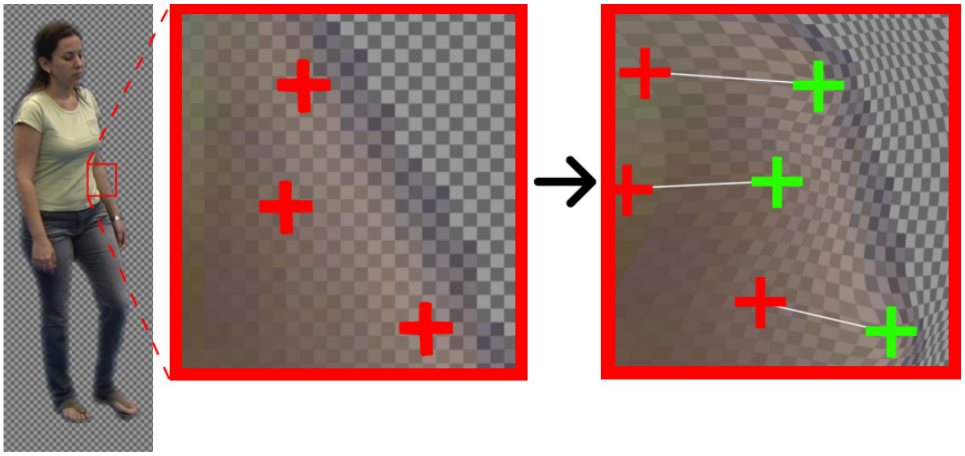
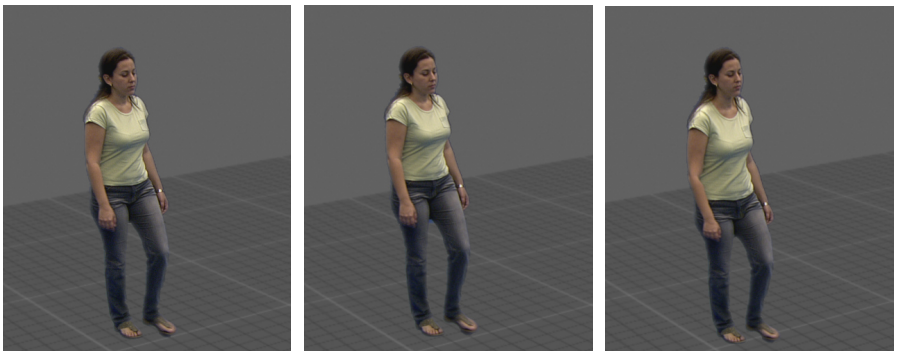


**Figure 4.6:** Illustration of the MLS-based warping of the actor's shape. The zoomed in region shows the projected deformation constraints in the *source model* configuration (left), and in the *target model* configuration (right). The red points show the source constraint positions, the green points the target positions. The image is warped to fulfill the target constraints.

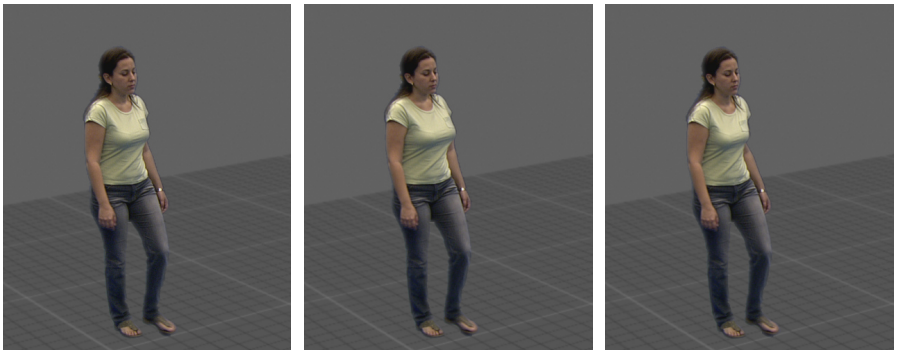
Our reshaping interface allows the user to generate a desired 3D target shape $\Lambda' = \Delta\Lambda + \Lambda$ from the estimated 3D source shape $\Lambda$ (remember that $\Lambda$ is constant in all frames after tracking has terminated). This change can be applied automatically to all the images of the sequence. In our system the user-selected 3D shape change provides the input for a meshless moving least squares (MLS) image deformation, which was introduced by [Müller et al. 2005; Schaefer, McPhail and Warren 2006] (see Section 4.8 for a discussion on why we selected this approach).

The 2D deformation constraints for MLS image deformation are generated by employing a sparse subset $\mathbb{S}$ of all surface vertices $\mathbf{v}_i$ of the body model. This set $\mathbb{S}$ is defined once manually for our morphable body model. We selected approx. 5 to 10 vertices per body part making sure that the resulting 2D MLS constraints are well distributed from all possible camera perspectives. This selection of a subset of vertices is done only once and then kept unchanged for all scenes. In the following, we illustrate the warping process using a single frame of video (Figure 4.6). To start with, each vertex in $\mathbb{S}$ is transformed from the standard model pose into the pose and shape of the *source body*, i.e., the model in the pose and shape as it was found by our tracking approach. Afterwards, the vertex is projected into the current camera image, resulting in the source 2D deformation point $\mathbf{s}_i$. Then, each subset vertex is transformed into the pose and shape of the *target body* - i.e., the body with the altered shape attributes - and projected in the camera image to obtain the target 2D deformation points $\mathbf{t}_i$:

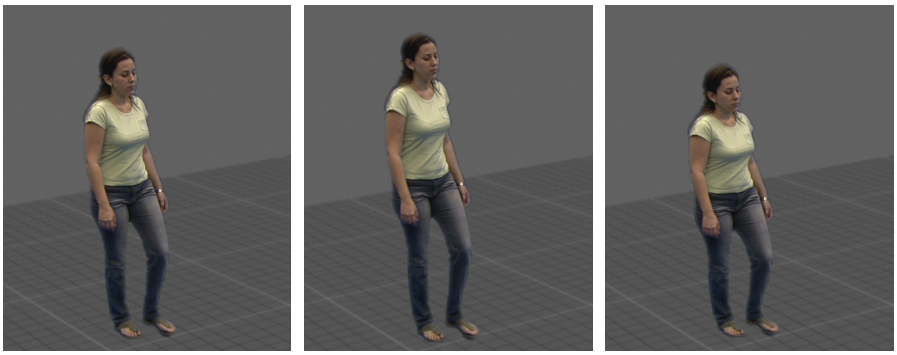
$$\begin{aligned} \mathbf{s}_i &= \mathcal{P}_t\left(\mathcal{T}(\Phi_t, \Lambda)\mathbf{v}_i\right) \\ \mathbf{t}_i &= \mathcal{P}_t\left(\mathcal{T}(\Phi_t, \Lambda')\mathbf{v}_i\right) \quad , \end{aligned} \tag{4.4}$$



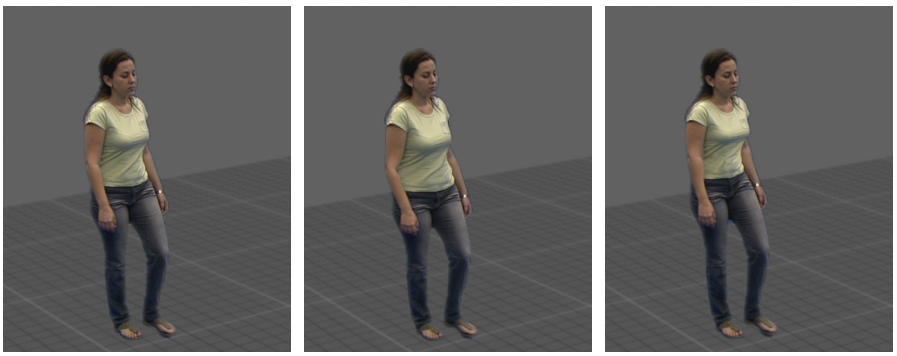
(a) original (b) leg length +2.5 cm (c) leg length -10.5 cm



(d) original (e) breast girth +13 cm (f) breast girth -6 cm



(g) original (h) height +15 cm (i) height -10 cm



(j) original (k) waist girth +12 cm (l) waist girth -5 cm

**Figure 4.7:** A variety of reshaping results obtained by modifying several shape attributes of the same actor.

where $\mathcal{P}_t$ denotes the projection in the current camera image at time $t$.

Given the deformation constraints $\mathbf{s}_i \rightarrow \mathbf{t}_i$, MLS deformation finds for each pixel $\mathbf{x}$ in the image the optimal 2D transformation $\mathcal{M}_\mathbf{x}$ to transform the pixel to its new location $\mathbf{x}' = \mathcal{M}_\mathbf{x}(\mathbf{x})$. Thereby, the following cost function is minimized:

$$\underset{\mathcal{M}_\mathbf{x}}{\arg\min} \quad \sum_{\mathbf{s}_i,\mathbf{t}_i \in \mathbb{S}} \frac{1}{|\mathbf{x}-\mathbf{s}_i|^2} (\mathcal{M}_\mathbf{x}(\mathbf{s}_i) - \mathbf{t}_i)^2 \quad . \tag{4.5}$$

The closed-form solution to this minimization problem is given in [Müller et al. 2005]. Similar as in [Ritschel et al. 2009], our system calculates the optimal 2D deformation in parallel for all pixels of the image using a fragment shader on the GPU. This allows the user of the reshaping interface to have an immediate *What You See Is What You Get*-feedback when a semantic shape attribute is changed. In practice, the user decides on the appropriate reshaping parameters by inspecting a single frame of video (typically the first one) in our interface. Figure 4.7 shows a variety of attribute modifications on the same actor. Once the user is satisfied with the new shape, the warping procedure for the entire sequence is started with a click of a button.

## 4.5 Results

We performed a wide variety of shape edits on actors from three different video sequences: **1)** a monocular sequence from the TV series Baywatch showing a man jogging on the beach (DVD quality, resolution: $720 \times 576$, 25 fps, duration 7 s), Figure 4.10; **2)** a monocular sequence showing a male basketball player (resolution: $1920 \times 1080$, 50 fps, duration 8 s), Figure 4.9; **3)** a multi-view video sequence kindly provided by the University of Surrey[2] showing a female actor walking/sitting down in a studio (8 HD video cameras, 25 fps, blue screen background, duration 5 s), Figure 4.7. More details can be found on the project website[3] or the supplemental video[4]. The sequences thus cover a wide range of motions, camera angles, picture formats, and real and synthetic backgrounds. The multi-view video sequence was tracked fully-automatically. In the monocular sequences, on average 1 in 39 frames needed manual user intervention, for instance the specification of some additional locations to be tracked. In neither case more than 5 minutes of user interaction were necessary. In the single-view sequences, the actor is segmented from the background using off-the-shelf tools, which takes on average 20 s per frame. All camera views in the multi-view sequence are chroma-keyed automatically.

The result figures show that we are able to perform a large range of semantically guided body reshaping operations on video data of many different formats that are typical in movie and video production. Figure 4.7 illustrates nicely the effect of

[2] http://kahlan.eps.surrey.ac.uk/i3dpost_action/

[3] http://resources.mpi-inf.mpg.de/MovieReshape/

[4] https://www.youtube.com/watch?v=zXSj4pcl9Ao

(a) original (b) muscularity +10% (c) muscularity +20% (d) muscularity +30%

**Figure 4.8:** Gradual increase of the muscularity of the Baywatch actor from his original shape (shown at the left).

the modification of individual shape attributes of the same individual. In all cases, the resulting edits are highly realistic. In the Baywatch sequence in Figure 4.10 we increased the muscularity of the actor by a significant amount. The final result looks highly convincing and consistent throughout the sequence. Figure 4.8 shows that gradual changes of the muscularity can be easily achieved. Figure 4.9 shows a basketball player filmed from a lateral angle. Our modification of the actor's waist girth looks very natural throughout the sequence, even for extreme edits that already lie beyond shape variations observed in reality. Overall, the modified actors look highly plausible and it is extremely hard to unveil them as video retouching results. Note that our edits are not only consistent over time, but also perspectively correct. Without an underlying 3D model such results would be hard to achieve.

Our results on the multi-view data (Figure 4.7 and supplemental video) illustrate that the system is also useful when applied to footage that has been captured under very controlled studio conditions. For instance, if scene compositing is the goal, an actor can be captured on set from a variety of pre-planned camera positions in front of a blue screen. Now, with our system the shape of the actor can be arbitrarily modified in any of the camera views, such that the director can decide during compositing if any shape edit is necessary. As an additional benefit, on multi-view data no manual intervention is needed, except the user input defining the edit. The accompanying video shows a few examples of combined shape editing and compositing with a rendered backdrop.

Using an unoptimized implementation on an Intel Core 2 Duo CPU, @3.0 GHz it takes around 9 s per frame to track the pose of the actor in a monocular sequence, and 22 s to do the same in the multi-view case. Note that tracking is only performed once for each sequence. In our reshaping tool, shape attributes can be modified in real-time, with immediate visual feedback given for the initial frame of the video. Generating the video with the new shape parameters, i.e., applying image-based warping to the entire video, takes approx. 20 ms per frame.

### 4.5.1 User Study

We evaluated our system in a user study. The goal of the study was to find out if small artifacts that may be introduced by our algorithm are noticeable by a human observer. We presented 30 participants the *Baywatch* video (shown in Figure 4.10 and in the supplemental video). Half of the participants were shown the original video and were asked to rate the amount of visible artifacts. The other half was shown our modified video, where the running man is rendered more muscular, and were asked the same question. The participants rated the amount of visible artifacts on a 7-point Likert scale, where 1 means no artifacts and 7 very disturbing artifacts. The first group, which watched the original video, rated the amount of visible artifacts on average with $2.733 \pm 1.22$, where $\pm$ denotes the standard deviation. Our modified video received only a slightly worse rating of $2.866 \pm 1.414$. This may indicate that slight artifacts are introduced by our method. We validated this assumption with a two-way analysis of variance (ANOVA). The null hypothesis that the means of the two groups are equal does results in a very high p-value of 0.709 and, consequently, such a null hypothesis should not be rejected. This leads us to the conclusion that the amount of artifacts introduced by our method is very low and, thus, the analysis does not show a significant effect to reject such a null hypothesis in our experiment (on the other hand, this does not show that such a null hypothesis is true and we have proven that there are no artifacts introduced by our method).

We then showed all 30 participants a side-by-side comparison of the original and the modified video and asked them if they could spot the difference. 28 out of 30 participants realized that we have made the running man more muscular, and only two participants thought that we changed something in the background. This indicates that our system is capable of achieving a noticeable reshaping result without introducing significant artifacts.

## 4.6 Additional Scenarios

### 4.6.1 Seeing a person in different shape

The approach could also be used to preview in real-time or after some off-line processing how a person moving in front of a video camera or other imaging device would look like when the body shape would be different. Here, the video of the person moving would be warped in real-time or off-line.

For instance, the approach could be used in a setup or measurement apparatus comprising one or several video cameras that record a person and another device that captures the motion of that person in real-time (or off-line). The statistical body model would then be fitted to the person using the video data and/or the the data from the additional sensor equipment. The additional sensor equipment could, for


(a) waist girth + 20 cm (b) Extreme reshaping - waist girth + 35 cm

**Figure 4.9:** Change of waist girth of a basketball player recorded with a single video camera - on the left, the waist girth was increased moderately; on the right the waist girth was increased way beyond a natural range, but still the deformation looks coherent and plausible.

instance, be a depth camera (Time-of-Flight or similar) or any other hardware for dynamic geometry capture. The additional hardware could also be another motion capture apparatus (e.g. optical motion capture system, electromagnetic motion capture, any form of motion capture suit etc.). The video data and / or the additional sensors would be used to track the motion of the person either in real-time or off-line. The proposed video morphing algorithm would then be applied to one or several of the video streams and show the person moving with an altered body shape.

One possible realization of this idea would be that the person moves in front of a measurement apparatus comprising a real-time depth sensor (e.g. depth camera, stereo camera, structured light scanner or similar apparatus delivering 2.5D or 3D geometry in real-time or at near-realtime frame rates). The statistical body model is fitted to the video and/or depth data. Here, the measured 2.5D or 3D data would serve as additional information that can be used by the tracking and model fitting procedures to achieve higher robustness (e.g. the 2.5D data could become part of the error function Eq. (4.1), but could also be used in other ways by the model fitting and tracking procedure to determine pose and shape parameters). The motion of the model is tracked in real-time from the video and / or depth data. The user can see himself in the video stream in real-time while the body model-based video warping is applied in real-time.

The above described realizations could also be applied to several people that are jointly captured by one or several video cameras and/or the additional sensor equipment described above.

### 4.6.2 Warping images

The statistical model-based warping approach can also be used to warp images of people. For instance, with an apparatus as described in Section 4.6.1), one could measure the pose and body shape of any person standing in front of the setup. The statistical model could also be fitted to any single image of a person, e.g. any image from a catalogue showing a person in a certain type of apparel. The motion and shape parameters of the person standing in front of the camera could now be transferred to the model fitted to any of the images. The person in the image could now be warped to match the proportions of the person in front of the sensor setup. In addition, the motion of the person in front of the sensor setup could be transferred to the image, by making the model fitted to the image imitate the motion of the person in front of the setup.

As before, also this idea could be realized in an on-line or an off-line scenario.

### 4.6.3 Application in Fitness Motivation Video Generation

The described method could be used to produce motivational image or video material to motivate people to achieve a certain goal regarding their body shape or fitness. Since our method can quantitatively specify changes in human body parameters (e.g. 10 kg more weight), a person could preview how he or she would look like after a weight loss, muscle gain or other changes of physical attributes.

### 4.6.4 Applications in Movie Production

For some movie productions, actors are required to alter their physical appearance, e.g. by training to gain more muscles, or by losing or gaining weight. With our approach, the actor would not have to go through these changes, as they can be physically demanding. Our approach can simulate the desired appearance of the actor on screen, even if his true body shape and proportions do not match the desired look.

It is also feasible to apply very strong changes to the attributes of actors, e.g. by turning one actor into a dwarf, and another actor into a giant, even though they are actually of similar height in reality.

### 4.6.5 Applications in Productions of Advertisements

Different cultures or groups of people may have different preferences regarding body shapes. Often an advertisement video is specifically targeted to one cultural environment. With our proposed algorithm a commercial or other type of promotional

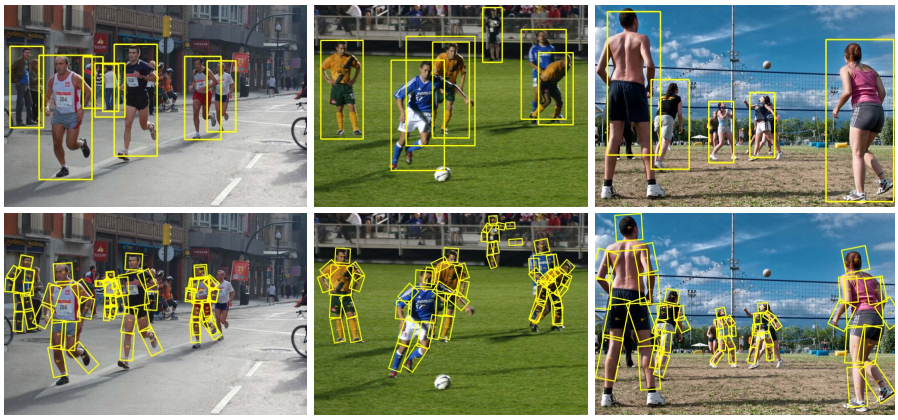


**Figure 4.10:** Sample detections (top) and pose estimates (bottom) of multiple articulated people obtained with our model trained on images from our new data generation method.

video could be filmed once, and different version with differently reshaped actors could be produces afterwards to meet the expectations of different target audiences.

## 4.7 Extending Existing Training Sets for Articulated Human Detection and Pose Estimation

State-of-the-art methods for human detection and pose estimation require many training samples for best performance. While large, manually collected datasets exist, the captured variations w.r.t. appearance, shape and pose are often uncontrolled thus limiting the overall performance. In order to overcome this limitation we propose a new technique to extend an existing training set that allows to explicitly control pose and shape variations. For this we build on recent advances in computer graphics to generate samples with realistic appearance and background while modifying body shape and pose. We validate the effectiveness of our approach on the task of articulated human detection *and* articulated pose estimation. We report close to state of the art results on the popular Image Parsing [Ramanan 2006] human pose estimation benchmark and demonstrate superior performance for articulated human detection. In addition we define a new challenge of combined articulated human detection and pose estimation in real-world scenes.

In this work we are interested in the challenging problem of articulated people detection *and* pose estimation in challenging real-world scenes. In order to achieve this goal (e.g. illustrated in Figure 4.10), we advance the state of the art in several ways. As a first contribution, we propose a novel method for automatic generation of multiple training examples from an arbitrary set of images with annotated human body poses. We use a 3D human shape model [Hasler et al. 2009] to produce a set of realistic shape deformations of person's appearance, and combine them with motion capture data to produce a set of feasible pose changes. This allows us to generate realistically looking training images of people where we have full control

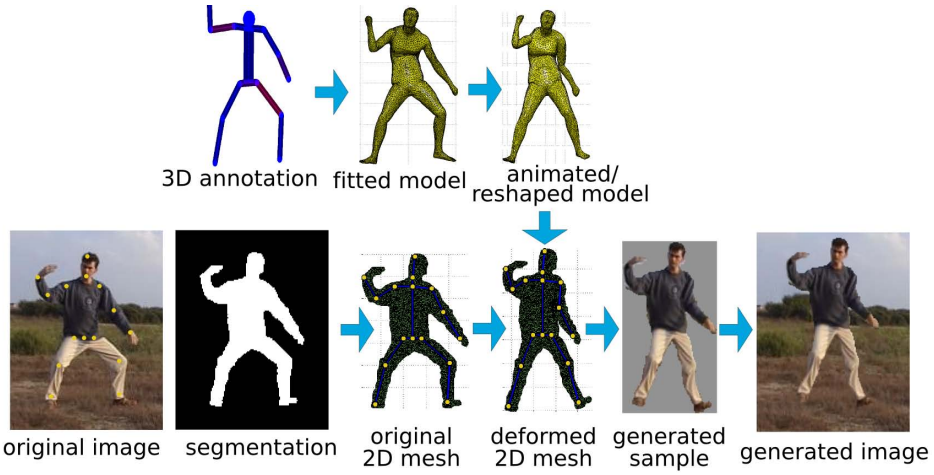



**Figure 4.11:** Overview of our novel data generation method.

over the shape and pose variations.

As a second contribution, we evaluate our data generation method on the task of articulated human detection and on the task of human pose estimation.

### 4.7.1 Generation of Novel Training Examples

To improve both articulated people detection and pose estimation we aim to generate training images with full control over pose and shape variations. Figure 4.11 gives an overview of our novel data generation process consisting of three stages. Starting from approximate 3D pose annotations we first recover the parameters of the 3D human shape model [Hasler et al. 2009]. The body shape is then modified by *reshaping* and *animating*. Reshaping changes the shape parameters according to the learned generative 3D human shape model and animating changes the underlying body skeleton. Given the new reshaped and/or animated 3D body shape we backproject it into the image and morph the segmentation of the person. To that end we employ the linear blend skinning procedure with bounded biharmonic weights described in [Jacobson et al. 2011]. The following describes these steps in more detail.

### 4.7.2 Data annotation

For each subject in the training set we manually provide a 3D pose and a semi-automatic segmentation of the person. The 3D pose is obtained using the annotation tool introduced in [Bourdev and Malik 2009]. The pose is used later to resolve the depth ambiguities which otherwise arise when fitting the 3D human shape model to 2D observations. The initial segmentation is obtained with GrabCut [Rother, Kolmogorov and Blake 2004] which we automatically initialize using annotated

2D joint positions and projected 3D shape from the fitted shape model (see below). While this procedure already produces reasonable results, segmented images often require user interaction to refine the segmentation due to low resolution, poor contrast and bad lighting. We use the segmentation to compute a 2D image mesh which is then deformed to change human shape and pose.

### 4.7.3 3D human shape recovery and animation

**Model fitting.** Having an annotated 3D pose allows to resolve the depth ambiguity while fitting the 3D shape model's kinematic skeleton to a 2D image. We re-target the skeleton to an annotated 3D pose by computing inverse kinematics through minimizing the Euclidean distance between a set of corresponding 3D joint positions, namely left/right ankles, knees, hips, wrists and elbows, upper neck and head. We use a constrained optimization based on the iterative interior point method. Optimization is done in shape and pose parameters space. Obtaining a good fit of the skeleton is essential for the rest of our data generation process and can significantly influence the realism of generated images. The fitting dependents on the flexibility of the kinematic skeleton and also on how well the corresponding 3D joint positions match. We thus do not include shoulders, pelvis and thorax joints into the objective function as these tend to have different positions in the annotated 3D pose and the 3D model's kinematic skeleton.

**Varying model shape and pose.** After fitting the skeleton we vary the 3D shape and pose parameters. To change the shape we randomly sample from the underlying 3D human shape distribution. For 3D shape animation we require a database of poses. To that end we re-targeted the shape model's kinematic skeleton to over 280,000 of highly articulated poses from freely available mocap data[5]. To do so, we fix the bone lengths of the mocap skeleton to be the same as for the shape model's skeleton and compute inverse kinematics by optimizing over global rotation, translation and pose parameters only, which reduces the search space and produces better results. To animate the fitted skeleton we use the nearest re-targeted poses with an average joint distance of less than 90 mm. Informal experiments showed that going further away from the fitted pose may result in unrealistically looking generated images.

### 4.7.4 Generation of novel images

After shape and pose changes are applied to the fitted 3D shape model, we project its 3D joint positions into the image and move 2D annotated joints towards corresponding projected joints. This results in a smooth 2D mesh deformations described

[5]CMU MoCap Database `http://mocap.cs.cmu.edu/`

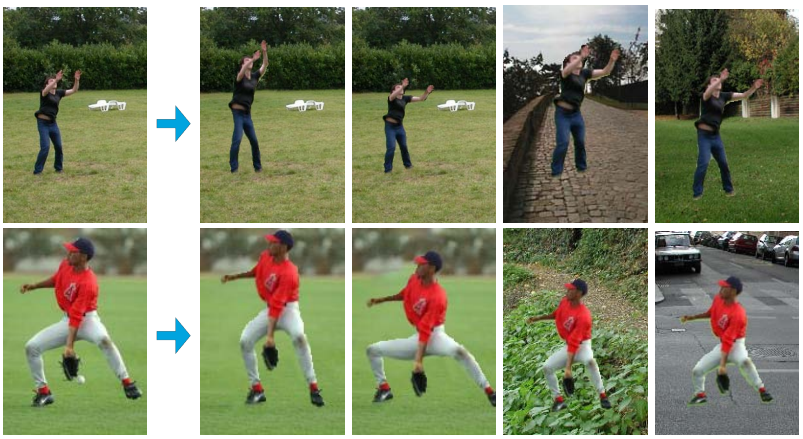



**Figure 4.12:** Examples of automatically generated novel images: (a) original image and (b) animated and reshaped synthetic samples with different backgrounds. Note the realism of the generated samples.

by linear blend skinning [Jacobson et al. 2011]. We only animate "dangling" arms and legs, and do not deform occluded or occluding limbs as this leads to unrealistic deformations.

To obtain a final training sample we render the deformed 2D mesh into a photo-realistically looking individual by reusing the original appearance of the person. Finally we combine the rendered subject with the background. We either replace the original person with the generated one by first removing the original person from the image using a commercial implementation of [Barnes et al. 2009], or embed the generated sample at a random place of a new people-free image. Figure 4.12 shows original images from the "Image Parsing" set and automatically generated novel images with animated and reshaped humans and different types of backgrounds.

## 4.8 Discussion

We demonstrated that our approach can modify the body shape of actors in videos extremely realistically.

Pixel-accurate tracking is hard to achieve, especially in monocular sequences. Therefore, we refrain from using a 3D model, which could be textured with the original video frame, for rendering the reshaped human. This would inevitably lead to noticeable artifacts. In contrast, our 2D image deformation that is guided by the 3D model is robust against small tracking errors and still produces perspectively correct warps.

Nonetheless, our approach is subject to a few limitations. If the pose tracking was sub-optimal, deformation constraints may be placed very close to or in the

scene background. In this case, the image deformation applied to the actor may propagate into the background leading to a halo-like warp. When the person's shape is extremely enlarged, distortions may become noticeable in the background (Figure 4.13). Similarly, when the person's apparent size is strongly reduced, the background is warped to fill the whole, whereas another option would be a spatio-temporal inpainting of the disocclusions. However, as confirmed in the user study, we found out that for a normal range of edits, these effects are hardly noticeable. In future, we plan to include inpainting functionality and apply a more advanced contour tracking and automatic segmentation approach. Figure 4.13(c) shows an example, where the shape manipulation enlarges the silhouette of the person. In that case it would be feasible to segment the person in the foreground, deform it, and overlay it with the original frame. This way, background distortions could be prevented. However, this alternative method may lead to even more objectionable artifacts, in particular if the segmentation is not accurate since the model boundary did not exactly coincide with the person's silhouette. As a consequence, we currently always employ MLS-based global image warping.

Another problematic situation arises when limbs are occluding other parts of the body. In this case the deformation of the occluded body part is also applied to the limbs, which is an undesired artifact. In practice the effect is not very noticeable for shape modifications in a normal range.

(a) (b) (c)

**Figure 4.13:** MLS-based image warping compared to segmentation-based deformation. (a) Original Image, (b) Deformation using MLS-based image warping. One can notice slight artifacts in the background when the human deformation is too strong, e.g. the straight edge of the basket ball court appears curved. (c) Covering the background with the modified image of the segmented human often produces more objectionable artifacts, such as a double arm, double legs or shoes.

While our system works for people dressed in normal apparel, our approach might face difficulties when people wear very wide clothing, such as a wavy skirt or long coat. In such cases, automatic pose tracking would fail. In addition, our warping scheme may not lead to plausible reshaping results that reflect the expected deformation of wide apparel. Also, shape edits often leads to corresponding changes in skeletal dimensions. When editing a video, this might make motion retargeting necessary in order to preserve a natural motion (e.g. to prevent foot skating). However, for most attribute dimensions this plays no strong role and even a modification of the leg length of an actor within certain bounds does not lead to noticeable gait errors.

Finally, our approach is currently not fully automatic. For segmentation of monocular video we heavily rely on commercial tools that may require manual intervention. However, we believe that the amount of user interaction required in order to make ill-posed monocular tracking feasible is acceptable, given the ability to perform previously unseen shape edits in videos.

# 5

# Data-driven 3D Modeling

Figure 5.1: A blend between a bike and a motor-bike with different structure, produced without user intervention by our technique.

## 5.1 Introduction

Easy-to-use shape modeling systems that produce custom detailed 3D models are hard to come by. Professional shape modeling tools are difficult to master, and detailed shapes take a long time to create. Thus, many non-expert users resort to simply choosing the most suitable existing 3D model from the increasing number of databases that are available on the Internet. Yet, it is often the case that no model in the database is entirely suitable, or that the user is looking for a custom model.

This work presents a system that can be used to synthesize new 3D models from a database of many-part shapes. The system is well suited for non-expert users because a blend between two database shapes, as shown Figure 5.1, can be controlled via a single slider. Professional users can employ the system to conveniently and quickly generate a large number of shape variations, e. g., it is possible to produce a crowd of hundreds of re-combined individual robots from only a few database shapes. Mixing shapes of different classes could help artists to brainstorm new shapes, e. g., by mixing a boat and an airplane to produce a fast-looking boat (cp. Figure 5.12).

The system aims at blending between models that are highly dissimilar from a conventional geometry processing point of view; they can have different numbers

of parts, different mesh connectivity, and different topological structure. Our key simplification is to avoid varying the geometry inside individual parts that constitute the shape. Instead, we segment the database models into parts and synthesize new models by recombining these parts.

The key challenge of part-based recombination is to minimize the synthesis of undesirable variations. The number of possible models that can be synthesized from given parts grows combinatorially, and most such models are undesirable. The key assumption behind our system is that preservation of symmetries and contacts found in the source models can increase the desirability of the synthesized models. This constrains the models that the system generates, leading to more visually pleasing blends.

Our system processes a database by segmenting the shapes into parts and computing symmetries and contacts between these parts. Contacts between the parts describe the adjacency structure of the shape. The adjacency structure is employed during the creation of a hierarchy that groups connected parts in a coarse-to-fine manner. At runtime, given two database shapes that the user wishes to blend, the system performs hierarchical matching between the shapes, re-mixing parts that have similar positional information in the nodes of the hierarchy. The computed matching is used to interpolate the adjacency structure of the models. Individual parts are exchanged during the interpolation and are positioned by a mass-spring system that enforces contacts.This produces new models that incorporate parts from given shapes yet vary in appearance.

## 5.2 Shape Analysis and Synthesis

Our approach consists of two phases discussed in this section. The offline phase is an analysis of a database of many 3D objects leading to a representation of shape-part relationships based on the hierarchical structure and contacts between parts. During the online phase, this representation is then used to synthesize new shapes from parts with relations similar to those in the database.

### 5.2.1 Shape Analysis

Shape analysis is used to find the relations between parts that constitute a shape. We start from $\mathcal{S} := \{S_i | S_i \in \mathcal{M}, i = 1, \ldots, n^s\}$, where $\mathcal{M}$ is the set of $n^s$ shapes $S_i$ in the database. Each shape $S_i$ is represented as a polygonal mesh. Our database currently comprises 280 different man-made objects, providing no symmetry or hierarchy information, which are taken from 3D model repositories on the Internet. These models typically have different scales, but it is a prerequisite that they have a consistent alignment to the global coordinate axes (as is typically the case for 3D models from Internet repositories). In particular, all 280 of our models have a

consistent upright orientation.

As pre-processing pipeline as explained in Section 2.2, is run on every shape in the database independently. It consists of segmentation, contact analysis, symmetry detection, and hierarchy generation (see Figure 5.2). This section provides the details of every step.

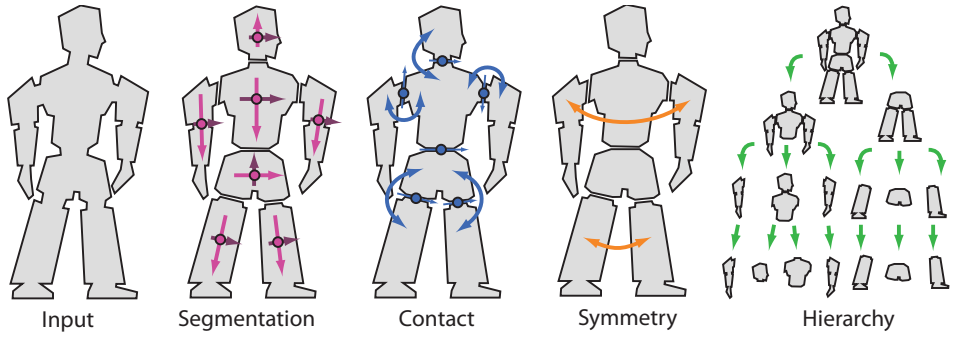



**Figure 5.2:** After segmenting the input shape, we detect symmetry and contacts on multiple levels of a shape hierarchy.

**Segmentation** The $i$-th shape $S_i$ is decomposed into $n_i^{\mathrm{p}}$ parts $S_i = \bigcup_{j=1}^{j=n_i^{\mathrm{p}}} P_{i,j}$ which are again polygonal meshes. In our case, segments are connected components of the polygonal input mesh, which are generated by region growing.

Next, every part $P_{i,j}$ is re-sampled to a point cloud $\bar{P}_{i,j}$ for further processing. The individual points of $\bar{P}_{i,j}$ are placed on the surface in such a way, that their distance is roughly equal (blue noise). Now, a principal component analysis (PCA) of $\bar{P}_{i,j}$ is performed, which provides a transformation $\mathsf{T}_{i,j}$ from the global into the local coordinate system of the part. A point $\mathbf{p}'$ in the local coordinate system is given by a transformation $\mathsf{T}_{i,j}$, which combines a translation, a rotation and a scaling: $\mathbf{p}' = \mathsf{T}_{i,j}\mathbf{p} = \mathsf{S}_{i,j}\mathsf{R}_{i,j}(\mathbf{p} - \mathbf{c}_{i,j})$. The geometric center of the part's point cloud in the world coordinate system defines the center $\mathbf{c}_{i,j}$ of the local coordinate system. The local $3 \times 3$ rotation matrix $\mathsf{R}_{i,j}$ is given by the three PCA basis vectors and defines the local rotation axes. The diagonal $3 \times 3$ matrix $\mathsf{S}_{i,j} = \mathrm{diag}(1/s_{\mathrm{x}}, 1/s_{\mathrm{y}}, 1/s_{\mathrm{z}})$ describes the local non-uniform inverse scaling using the three singular-values $\mathbf{s}_{i,j} = (s_{\mathrm{x}}, s_{\mathrm{y}}, s_{\mathrm{z}})^{\top}$.

**Contact Analysis.** During contact analysis all intersections of all parts for a shape are found. For each part $P_{i,j}$ of shape $S_i$ it is evaluated, if it is in contact with another part $P_{i,k}$. We call the subset of points in the point cloud $\bar{P}_{i,j}$ for which a point with a distance of less than 0.1 % of the bounding box diameter exists in $\bar{P}_{i,k}$ the *contact* $C_{i,j,k}$ of part $j$ and $k$. In practice, an axis-aligned bounding box tree [van den Bergen 1998] on all points of shape $S_i$ is used to compute the set of contact points efficiently. The set of contacts describes the adjacency structure of a shape.

In summary, the above two steps produce a segmentation of each shape into parts,

as well as a list of contacts between the parts of a shape. An example is shown in Figure 5.3.

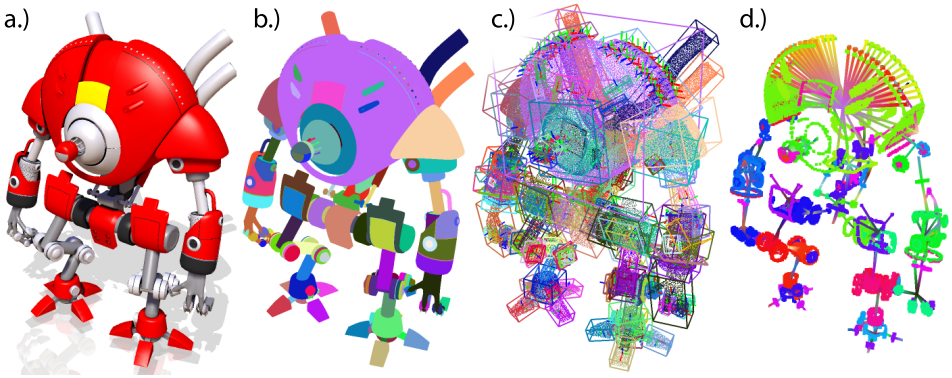



Figure 5.3: *a.)* Shape, *b.)* Segmentation, *c.)* PCA, *d.)* Contacts.

**Symmetry Detection.** Next, the dominant global symmetry transformation $\mathsf{H}_i$ (a reflection, rotation, or translation) for the $i$-th shape is found using a RANSAC approach [Berner et al. 2008]. The approach randomly samples a number of potential symmetry transformations. In every trial, one symmetry transformation candidate $\mathsf{K}$ is generated. Then, the support $\alpha(\mathsf{K})$ of all parts for this symmetry candidate is computed. To this end, the center $\mathbf{c}_{i,j}$ of every part $P_{i,j}$ is mapped to $\mathbf{p}'_{i,j} = \mathsf{K}\mathbf{p}_{i,j}$, and if a matching part $P_{i,j'}$ is found at $\mathbf{p}'$, a support counter is incremented. Two parts match if their eigenvalues $\mathbf{s}_{i,j}$ and $\mathbf{s}_{i,j'}$ are similar, i. e., they are of similar shape. After all trials, the symmetry with the highest support count $\mathsf{H}_i = \arg\max_{\mathsf{K}} \alpha(\mathsf{K})$ is assumed to be the dominant symmetry.

To generate candidate transformations, the following procedure is used: for reflective symmetry, two random parts $P_{i,j}$ and $P_{i,k}$ are selected. The difference vector $\mathbf{d}_{i,j,k} = \mathbf{c}_{i,k} - \mathbf{c}_{i,j}$ between the part centers $\mathbf{c}_{i,j}$ and $\mathbf{c}_{i,k}$ defines the normal of a reflective symmetry plane, and a reference point on the plane is given by $(0.5\,\mathbf{d}_{i,j,k} + \mathbf{c}_{i,j})$. Translational symmetries are found by directly using $\mathbf{d}_{i,j,k}$ as a translational offset. For rotations, a third part $P_{i,l}$ is selected and a circle is fitted to the center of all three parts defining a rotation around a point by an angle.

In practice, we first compute the best reflective symmetry. If its support is below a threshold of 80 %, we assume no reflective symmetry and compute the best translational symmetry. If the support of this symmetry is below 80 % as well, the best rotational symmetry is computed. If rotational symmetry is supported by less than 80 %, no symmetry is assumed.

The result of this step is the one most dominant global symmetry for each shape (if present). Most shapes in our database exhibit a dominant reflective symmetry, which is therefore preferred by the described approach over other forms of symmetry. Since the manually modeled 3D objects are almost noise-free, the detection of symmetries is typically very robust.

**Hierarchy.** Finally, a hierarchy is generated for each shape $S_i$. This hierarchy is constructed in a coarse-to-fine manner and has approximately $\log(n_i^{\mathrm{p}})$ levels. On each level, parts are grouped into hierarchy nodes $N$, which know their children and store this information. The $j$-th node of the $i$-th shape at level $k$ is denoted as $N_{i,j,k}$. On each level each node computes and stores its corresponding eigen-transformation $\mathsf{T}_{i,j,k}$ as well as its symmetry transformation $\mathsf{H}_{i,j,k}$, if it exists.

On the coarsest level, $k = 0$, all $n_i^{\mathrm{p}}$ parts are added to a single root node $N_{i,0,0}$. As the single root node on level 0 comprises all the parts of the shape, its eigen-transformation $\mathsf{T}_{i,0,0}$ corresponds to the eigen-transformation of the complete shape $S_i$ and its symmetry transformation is given by the dominant global symmetry.

When going from a coarser level $(k-1)$ to a finer level $k$, for each parent node $P_{i,j,k-1}$ on the coarser level, two child nodes on the finer level are generated. Only those parts that belong to the parent node are now split into two sets and those sets are assigned to the two child nodes on the finer level. If a symmetry was detected, the splitting into children takes this information into account. For reflective symmetry, the reflective symmetry plane splits the parts of the parent node into two sets that are assigned to the two children. In a similar fashion, splitting planes can be defined for translational and rotational symmetry, e. g., for translational symmetry the splitting plane is located halfway on the translational offset vector. A third child is added in those cases where the centroid of a part is located in close proximity to the splitting plane. The splitting operation is successful if at least two child nodes have at least one part assigned. If the splitting based on the symmetry information is not successful or if no symmetry is available, the splitting plane $x = 0$ in the local coordinate system of the parent (defined by the parent's eigen-transformation $\mathsf{T}_{i,j,k-1}$) is used. If this splitting operation fails as well, the $y = 0$ and $z = 0$ planes are tested. If the operation is unsuccessful for any splitting plane, only a single child is generated for the parent node, where the child contains all the parts of the parent. Otherwise, if the splitting operation was successful, each child's corresponding eigen-transformation is computed and stored. Furthermore, the symmetry detection algorithm described in the last paragraph is applied to all the parts of the child node. This process is repeated for subsequent levels until a level is reached, where no further splitting operations can be performed. Figure 5.4 shows the hierarchy for an example shape.

**Enforcing Nodes without Disconnected Parts.** At this stage, the hierarchy generation does not take into account contact information between parts. Consequently, it can happen that a child node contains parts that are disconnected, i. e., that there is no connection path between the parts over one or multiple contacts. In this paragraph we describe an algorithm which ensures that each node $N$ of the hierarchy contains only non-disconnected parts. This algorithm is executed after the creation of each new level $k$ in the hierarchy. Let's assume that after its creation the level $k$ has $n_k^{\mathrm{N}}$ nodes.

In a *separating step*, a region-growing algorithm on the contact adjacency structure is used to decompose the parts of a child node on the current level $k$ into sets of

**Figure 5.4:** Generation of a hierarchy that takes symmetry into account: Symmetric parts are grouped to nodes on the same level. The coarsest level of the hierarchy is shown top left, the initial part segmentation at the bottom right.

non-disconnected parts. For each set, we generate a new child node in the current level that contains the parts of the set. Thus, the original child node may be replaced by multiple new ones. After the separating step is performed for all original child nodes, we execute a *merging step* that tries to merge the child nodes (in order to obtain a similar number of nodes as the number $n_k^{\mathrm{N}}$ of nodes on the current level $k$ in the original configuration). All the nodes of the current level are sorted by area, and the $n_k^{\mathrm{N}}$ largest nodes are kept. The other nodes are merged with a kept node with which they share at least one contact. If the merged nodes share contacts with multiple kept nodes, the smallest of the kept nodes is selected for the merge operation. During the merge operation, all parts from each merged node are added to the selected kept node and the merged nodes subsequently are deleted.

After the separating and merging step, we have the same number of nodes as before but it is ensured that each node only contains non-disconnected parts. There is, however, an exception. In cases where merged nodes are available that have very similar size as the smallest kept node (we used a threshold of 95 %), those nodes are also kept and not merged. This is necessary for the algorithm to produce the same results for different branches of the tree hierarchy because the selection by size would otherwise be arbitrary. Figure 5.5 shows an example of the enforcement of non-disconnected nodes.

All of the above analysis is pre-computed and serialized to disk. It will be used to synthesize new shapes in the next step.

### 5.2.2 Shape Synthesis

The synthesis of new shapes is performed online at interactive speed using the pre-computed analysis results obtained from the previous steps. It consists of three

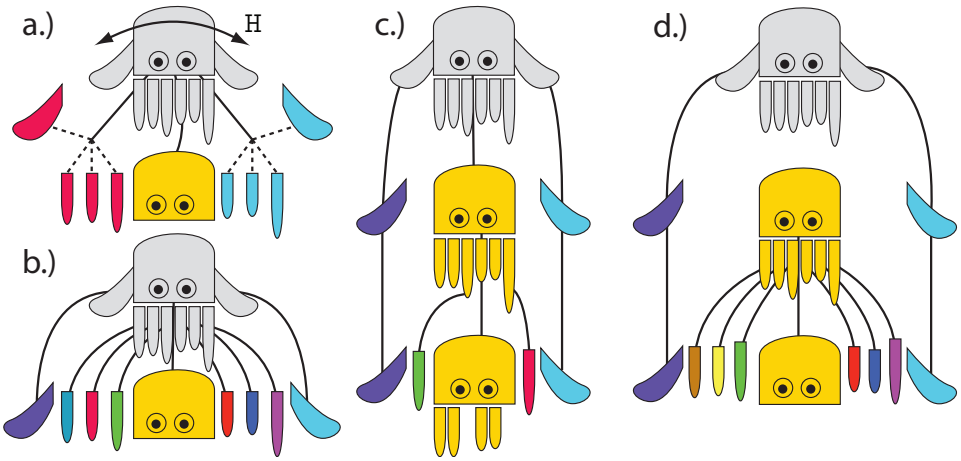



**Figure 5.5:** Enforcing non-disconnected nodes: *a.)* Split along the dominant (reflective) symmetry `H` during the creation of the hierarchy may result in disconnected nodes (dashed lines). *b.)* In the separating step disconnected nodes are split into individual non-disconnected nodes *c.)* The merging step ensures that there are only a low number of additional nodes for each level. However, this leads to arbitrary decomposition of tentacles on the second levels (if the tentacles have very similar sizes). *d.)* In this particular case our merging step make an exception and separates all tentacles, which have similar size, at the same level into separate nodes.

steps: shape matching, interpolation, and contact enforcement.

### Shape Matching

While all previous steps were performed on individual shapes, in this step, a matching between two shapes $S_1$ and $S_2$ is established.

In general, the number of parts is not the same for each shape; thus, a one-to-one mapping for parts cannot be established. This problem could be resolved to some extent by selecting a hierarchy level for both shapes where the number of nodes on both sides are the same, and performing matching between nodes on those levels. The shapes in our database, however, typically have a very different structure, resulting in a different hierarchy. As an example, let's assume the source shape is a car and the target shape is a truck, as shown in Figure 5.6a. Both models have the same number of nodes, but the back of the car (yellow) has to be used as one of the truck's tires to generate a one-to-one mapping of nodes. Instead, we propose to rebuild a new hierarchy for the target shape that adapts to the observed segmentation of the source shape during matching (as can be seen in Figure 5.6b).

An overview of the proposed shape matching approach is given in Algorithm 1. In the beginning, a target hierarchy root node is generated. As illustrated in Figure 5.7, all target parts are assigned to the root node, which defines the new coarsest level ($k = 0$) of the target shape $S_2$. We then calculate the eigen-transformation $\mathsf{T}_{2,0,0}$ of the target root node. We also define that the source and the target root nodes are in


**Figure 5.6:** Re-grouping parts (represented by polygons) into nodes (represented by the same color) of a new hierarchy can help to find a better match between two shapes $S_1$ and $S_2$: *a.)* original hierarchy for both shapes; *b.)* updated hierarchy of the target shape $S_2$.

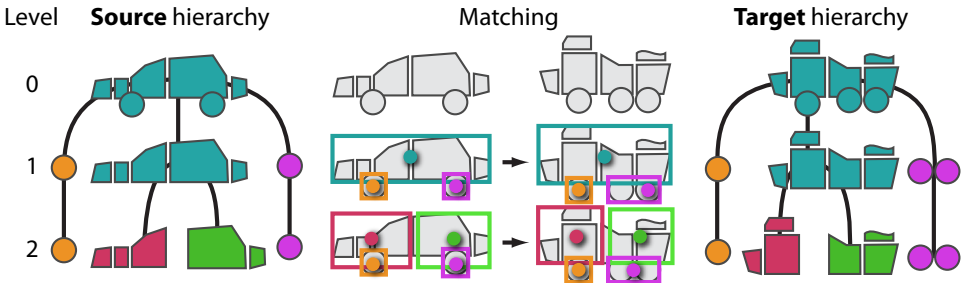



**Figure 5.7:** During shape matching the target hierarchy is re-generated. The structure of the source hierarchy is reproduced for the target shape (if possible). The one-to-one mapping of the nodes on each level directly defines the matching between the two shapes.

**Algorithm 1** Shape Matching

Build the target hierarchy root node
**for** all source hierarchy levels starting from level $k = 0$ **do**
**1.** Copy the child node structure of the current source level into the target (empty target child nodes);
**2.** Assign target parts to target child nodes by nearest neighbor matching, which compares the positions of the corresponding source child nodes with the positions of the target parts (all given in the local coordinate system of their parent node);
**3.** If any target child node is empty, remove it from the target hierarchy and merge the corresponding source child node with another source child node;
**4.** Enforce nodes without disconnected parts in the target hierarchy;
**end for**

correspondence, which means that they match onto each other.

The algorithm now iterates over all levels of the source shape. We start from the coarsest level ($k = 0$) of the source shape and, in each iteration, the next finer source level ($k+1$) is processed. For each iteration, a corresponding target level is created, so that the source and target hierarchies ultimately have the same number of levels.

In order to generate a level $k$ of the hierarchy for the target shape $S_2$, the following steps are executed (cp. Algorithm 1): First, the same number of child nodes in the target hierarchy is generated as on the current level of the source shape. The child nodes in the source and in the target are defined to be in exact correspondence, i. e., the child nodes with the same indices match onto each other: $N_{1,j,k} \leftrightarrow N_{2,j,k}$. The nodes in the target shape are empty at the moment, i. e., they currently do not have assigned parts. The parent and child nodes in the target are now linked exactly the same, as the source parents are linked with their child nodes in the source hierarchy. This means each target parent links to the target child nodes with the same indices as its corresponding node on level $(k-1)$ in the source hierarchy links to its children.

In the second step, all parts of a target node are distributed among the children. Each target child node knows its corresponding source child node. The part assignment is based on nearest neighbor matching of the positions of a target part's center in the local coordinate system of the target parent, and the position of the corresponding source child node in the local coordinate system of the source parent. To be more explicit, let $\mathsf{T}_{2,p(j),k-1}$ be the eigen-transformation of the target parent node. The position of the part's center $\mathbf{c}_{2,j}$ in the local coordinate system of the parent is then calculated by $\mathbf{c}'_{2,j} = \mathsf{T}_{2,p(j),k-1}(\mathbf{c}_{2,j})$. Similarly, let $\mathsf{T}_{1,p(m),k-1}$ be the eigen-transformation of the parent of source child node $N_{1,m,k}$. The position of the source child node's center $\mathbf{c}_{1,m}$ in the local coordinate system of the source parent is given by $\mathbf{c}'_{1,m} = \mathsf{T}_{1,p(m),k-1}(\mathbf{c}_{1,m})$. During nearest neighbor-matching, target parts are assigned to the target child node with the smallest distance between the local position of the corresponding source child node $\mathbf{c}'_{1,m}$ and the local part position $\mathbf{c}'_{2,j}$ (cp. Figure 5.7). Afterwards, the eigen-transformation for each child node of the target is calculated.

In the third step, the special case where a target child node has no attributed target part is considered. Here, the target child node is removed from the target hierarchy. Consequently, to ensure an exact one-to-one mapping, the corresponding source node has to merge with the closest child node of its parent that still has at least one target part in the corresponding target node. As a result the source hierarchy is modified.

Similar to the shape analysis procedure, nodes in the target shape hierarchy might have disconnected components. Thus, in the fourth step, we run the exact same algorithm as in the shape analysis to ensure that parts in a target node are not disconnected, with one exception: it is always ensured that the number of nodes on the same level does not change (during the shape analysis, we allow to create more

nodes if nodes selected for merging have very similar sizes as kept nodes).

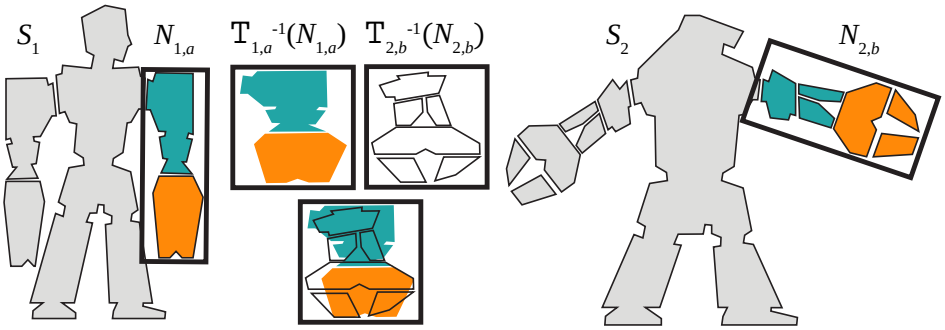



**Figure 5.8:** Matching parts inside their parent nodes $N_{1,a}$ and $N_{2,b}$ (black boxes) of two shapes $S_1$ and $S_2$ (left and right humanoid). The parts in each node are transformed into the unit cube by the inverse eigen-transformation $T_{1,a}^{-1}$ and $T_{2,b}^{-1}$ and matched to the nearest neighbor.

In summary, after the algorithm has processed all source levels, the source and target hierarchy have the same number of levels and the same number of corresponding nodes on each level; however, the individual nodes $N_{1,j,k}$ and $N_{2,j,k}$ may have different numbers of assigned parts. The matching between source and target shape is directly given by the correspondence between the nodes $N_{1,j,k} \leftrightarrow N_{2,j,k}$.

This algorithm only takes positional information into account; however, the positions are compared in the local coordinate systems of the parent node. Thus, the shape of the parts that were assigned to the parent play an important role, as those define the parent's local coordinate system. Let's assume we want to match the arms of two humanoid shapes as shown in Figure 5.8. If the matching on the previous level was successful, each part of the arm is now defined in its local coordinate system and part matching and conjoined re-grouping will return a reasonable result. This approach of conjoined matching and re-grouping is able to handle the difficult problem of matching parts that have different amounts of detail (like the arm in Figure 5.8). This is demonstrated in Figure 5.9 with two 3D aeroplane models from our database. The source shape, a lear jet, has no turbines on the wings, and thus a wing is modeled with only a few parts. The target shape, an MD-11, has turbines at the wing, and the complete wing (including the turbine) exhibits a large number of parts. As can be seen in the matching results of Figure 5.9, our algorithm handles this situation by just grouping the wing and the turbine of the target shape in a single node in order to adapt to the source shape.

This approach can be extended to not only consider spatial information when attributing a target part to a target node. For example, parts with similar shape or size can be assigned with higher likelihood; however, as the models in our database have rather different shapes, especially in the fine details, we resorted to taking only positional information into account.

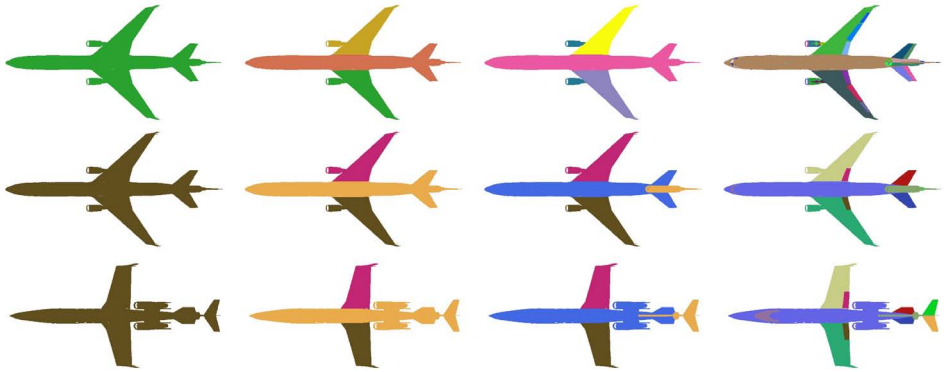


**Figure 5.9:** Result of the matching and conjoined re-grouping of nodes. The source shape is a Learjet and the target is a McDonnell Douglas MD-11; *Top row:* Input hierarchy of the target, *Left:* The three coarsest levels, *Right:* finest level; *Middle row:* resulting hierarchy of the target; *Bottom row:* resulting hierarchy of the source (same colors identify matching parts for the lower two rows).

### Shape Interpolation

Using the shape matching described previously, allows to interpolate shapes composed of parts, i. e., to generate „in-between" composition of parts into shapes.

**Linear Interpolation.** A new shape $S(w)$ can be generated depending on a *weight* parameter $w \in [0,1]$ to blend between two shapes $S_1$ and $S_2$. As it is customary for interpolations, it is clear that $S(0) = S_1$, and $S(1) = S_2$. The question is what would be expected, e.g., at $S(0.5)$?

In our approach, shapes are interpolated using the nodes on the finest level of the hierarchy. The employed shape matching ensures that matching nodes have the same index $j$ in the source shape $S_1$ and in the target shape $S_2$. During a blend, every part from $S_1$ should be replaced only once by a part from $S_2$ and never change back. This can be achieved by instantiating the first $w \cdot n_l^{\mathrm{N}}$ nodes (and all containing parts) from $S_2$ and all others from $S_1$:

$$P_j(w) = \begin{cases} P_{2,j} & \text{if } j < w \cdot n_l^{\mathrm{N}} \\ P_{1,j} & \text{else} \end{cases}$$

If two nodes in the source or in the target are symmetric, it is ensured that either both or neither are instantiated. The result depends now to some extent on which index $j$ was assigned to a node because, during a blend, a node with a small index is instantiated earlier for an increasing $w$. Thus, we have chosen to reassign the indices by sorting the corresponding nodes by the conjoined source and target node size. Obviously, the reassignment of indices must be done similarly in the source and the target hierarchy to ensure that the matching of nodes (which is encoded in the index) is not lost. Our chosen reassignment places the node with the largest

conjoined size in the middle of the blend, at $w = 0.5$, and node sizes are decreasing towards both sides $w = 0.0$ and $w = 1.0$. This choice typical gives visually pleasing blends. However, a user of our system can choose from several different sorting criteria, if desired.

**Incremental interpolation of multiple shapes.** Linear interpolation generates classic morphs between two shapes. Interpolation between more than two shapes is performed by executing multiple consecutive blends between two shapes. Starting from an existing shape $S_1$, this shape is modified by an incremental step of size $w_{1\to2}$ to become more similar to another shape $S_2$. This is achieved by simply exchanging parts in $S_1$ by a number of parts in $S_2$ proportional to $w_{1\to2}$, as described above. The blended shape $S_{1\to2}$ can then serve as a source shape for the next blend operation with a third shape $S_3$, and so on.

### Contact Enforcement

While the interpolation rule tells us which nodes from which model should be instantiated, the exact positioning of the node's parts in the new shape must still be optimized. A good placement can be found by enforcing the contacts between nodes as these were observed in the source and in the target shape. This may impose multiple conflicting positioning constraints and a consensus needs to be found. Nodes that were in contact in the source and target shape should be attached to each other in the interpolated shape as well (Figure 5.10). A node (from the source or from the target) usually tends to be in contact with its contact partner part that originated from the same shape. If this partner is unavailable, the node will try to enforce a contact with a node from the other shape that forms a match with the original contact partner. As shown in Figure 5.10, it will often occur that nodes have a mutual desire to connect to each other. These bi-directional contacts will be enforced in any case. If the wish to form a connection is only unidirectional, such contacts are only enforced if they are not in conflict with bidirectional contacts (cp. Figure 5.10).

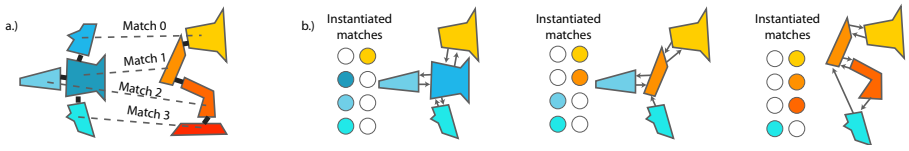



**Figure 5.10:** Finding contacts for interpolated shapes: *a.)* for the source shape and the target shape it is known which nodes are in contact by the shape analysis; *b.)* to generate an interpolated version between the source and the target shape, either the node from the source or from the target shape can be instantiated for each match. An instantiated node wants to connect to its contact partners that originated from the same shape; however, if a contact partner is not available, a node connects to the node of the other shape with which the original contact partner forms a match.

We employed a simple mass-spring system to enforce the contact constraints be-

tween nodes. This system is a set of masses with locations and a set of springs that connect masses. The system to enforce our constraints is set up as follows: one mass is created in the node's center and one for every contact to another node. A spring is then generated between every node's center and each of its contact points. The contact point of each node with a connected node is calculated by averaging all the contacts points of the individual part contacts $C_{i,j,k}$, where part $j$ is a member of the current node and part $k$ is a member of the connected node. The springs within a node want to keep their length and enforce that the distance between the center of a node and its node contacts is maintained. We then add zero-length springs between contacts of different nodes that should be enforced. These springs seek to reduce their length to zero and are consequently pulling nodes towards each other. The mass-spring setup for a simple example is shown in Figure 5.11. The mass-spring system is solved in a Jacobi fashion by successive over-relaxation [Müller et al. 2007]. Here, in each iteration, both masses connected by a spring are moved to make the spring come closer to its rest length.

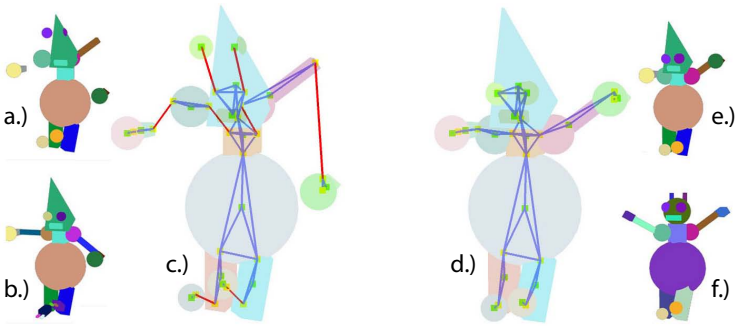



**Figure 5.11:** A simple example demonstrating the mass-spring contact enforcement: *a.)* interpolated shape without mass-spring, *b.)* source shape, *c.)* mass spring system, *d.)* solved mass spring system, *e.)* solved shape and *f.)* target shape.

Note that a mass-spring system (or an equivalent optimization procedure) is required for contact enforcement, as a simple top-down relation between nodes (and their parts) does not exist. The contacts between nodes do not, in general, form a tree but a graph. For shapes that have the structure of a tree, children would just need to follow their parents. While many man-made objects appear as trees on a coarse level, in their details they are indeed graphs.

## 5.3 Results

In this section, we present the results that were generated with our system. More results can bee seen in the supplemental video[1] or on the project website[2]. Users

[1] `https://www.youtube.com/watch?v=oHrBcyLzVzM`
[2] `http://resources.mpi-inf.mpg.de/3DModelRecombination/`

of our system can create new shapes by blending between database shapes. To this end, we offer a simple and very intuitive user interface. When the user clicks the *new* tool button, icons of available models in the database are shown. The shapes are organized into categories (e.g., “robots” or “ships”) in order to help browsing the database. The user then selects a shape that should be used as the starting point of the exploration process. To create a blend, the user clicks the *blend* tool button and selects the target shape of the blend operation. Now a single slider appears which lets the user control the blending process. The slider varies the weight $w$ of the linear interpolation between the source and the target shape (cp. Section 5.2.2). Figure 5.12 shows some results of such blending operations for different source and target shapes. The results shown here are blends for complex shapes with several hundreds of parts. More results for shapes with only a few parts and from different classes (e. g., tables, lamps, or cars) are given in the supplemental material. Due to the precomputed shape analysis, the blends can be performed at interactive rates with high-quality rendering feedback. Interactive user sessions are shown in the supplemental video. The two robots in the top row of Figure 5.12 both have $\approx 2.5$ M polygons. Recomputing their shape analysis takes approx. 10 s each, shape matching 720 ms, and the rest of the shape synthesis runs at 5 fps. Figure 5.13 shows a matrix of blend results for a weight of $w \approx 0.5$. The matrix contains blends for all combinations that are possible to generate between five different robots in our database. More matrices for different shapes are provided as supplemental material. Figure 5.14 shows a result for incremental interpolation of multiple shapes.

**Figure 5.12:** Results of the blend operation (far left source shape, far right target shape)

It is also possible to reproduce to some extent the functionality known from the *Shuf-*

**Figure 5.13:** Matrix of blend results between 5 different robot shapes in our database. The diagonal contains the original shapes, the off-diagonal the blends for a weight of $w \approx 0.5$. As described in Section 5.2.2 the blend is not symmetric.

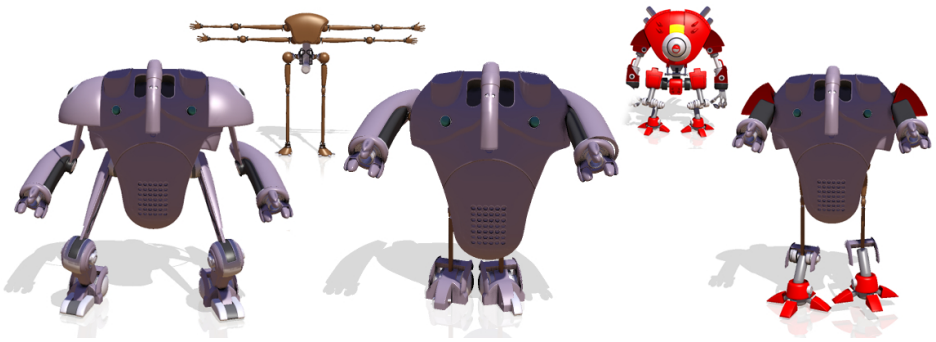


**Figure 5.14:** Blending three shapes: (from left to right) source shape, first target shape, intermediate result, second target shape, final result that contains parts from all three inputs.

*fler* system [Kraevoy, Julius and Sheffer 2007], which allows the user to control which part of the current shape is replaced. We call this functionality the *object brush* as it is possible to paint the parts of a selected new shape onto the current shape. In Figure 5.15 the result of such a brush operation is shown. The algorithms employed during the brush operation are exactly the same as those for blending; the only difference is that the user has full control over which node $j$ is flipped from the source to the target, i. e., when the user clicks on a source node the algorithm simply instantiates the corresponding target node. We evaluated our system with

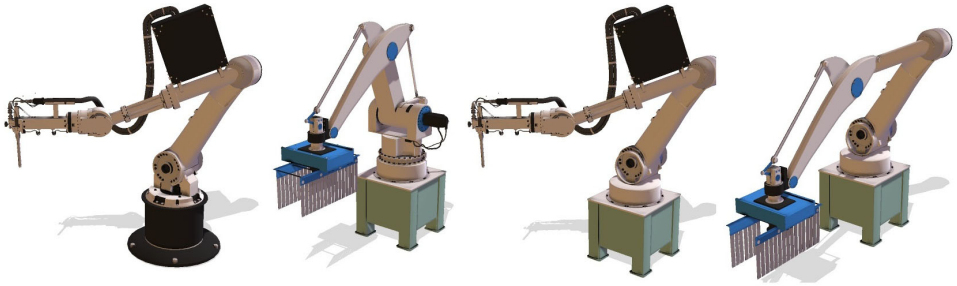


**Figure 5.15:** Exchanging parts of a shape using the object brush: (from left to right) the source shape, the target shape, the user clicked on the black base of the source shape, the user then clicked on the extended arm.

an informal user study with 14 computer science students who had never used the system before. On average the participants needed 01:41 min:sec to create a new shape with the blend tool, and it took 01:37 min:sec to reproduce a given result with the object brush tool. When asked if our system was helpful to solve the given task, the average rating over three tasks was 6.05 on a 7-point Likert scale (where 1 is worst and 7 is best). Details of the user study are provided as supplemental material.

## 5.4 Discussion

Not all the shapes generated by our system are useful or visually pleasant. Nevertheless, most of the results are of high quality, especially when it is taken into account that there was no manual intervention in the complete processing pipeline. As can be verified in Figure 5.13, it is possible to generate a large number of blended shapes that have similar quality as those that come directly from the database.

Currently, we only provide a very high-level user-interface with very easy-to-use tools; however, our system currently has no tools to repair a shape, e.g., when an undesired placing of a part occurs. One option would be to provide our system as a plug-in for a conventional modeling package. A falsely placed part can then be corrected easily with the editing tools provided by the conventional modeling package.

Most of the time, unpleasant or strange-looking recombinations occur if the structure of the source and the target shape are very different or objects of different classes are combined, i. e., a morph between two boats does typically look more convincing than a morph between a boat and a plane (cp. Figure 5.12). Nevertheless, our system produces reasonable results even in those situations. This works only because our algorithm updates the resolution hierarchy during matching. Figure 5.16 compares our approach to spectral graph matching [Leordeanu 2009], which does not re-group parts to form a new resolution hierarchy. Spectral graph matching additionally allows pairwise scores between two matches. For example, a high score can be given to two matches if there is a contact between the two involved parts in the source shape and there is also a contact between the two involved parts in the target. Furthermore, it is possible to assign higher pairwise scores to two matches if two source parts are symmetric and the corresponding targets parts are symmetric as well. However, for our particular application, the algorithm presented above usually gives better results (a comparison is shown in Figure 5.16).

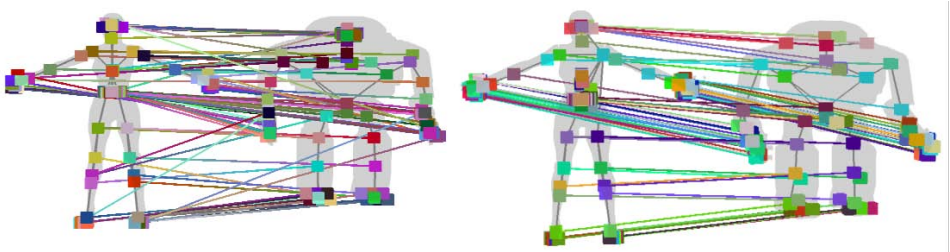


**Figure 5.16:** Graph matching between parts of two humanoid figures. *Left*: spectral graph matching. *Right*: our approach. Note that our approach prevents, e. g., parts of the foot from being mapped to the hand. The spectral graph matching finds a good global solution; however, as the resolution hierarchy is not updated, it cannot perform as well as an approach that combines matching and re-grouping of parts.

In comparison to Shuffler [Kraevoy, Julius and Sheffer 2007], where users specify

individual parts to be replaced, our system performs complete blends where all parts have to be subsequently exchanged. Furthermore, our approach differs in the employed shape segmentation and shape matching strategy. In the Shuffler approach segments are matched that have a similar midpoint graph distance and similar geometry. The goal of our method is to allow matching of parts that have rather different geometries (cp. Figure 5.13); consequently, our approach relies on positional (and contact) information instead.

The quality of the blended results depends on the way segments are grouped into shapes. If they are coupled loosely, with few contacts (e. g., limbs of robots, or features attached to the body of a vehicle), the approach works best. For too complex contact constraints, such as segments inside a car, the recombination might fail. Our approach does not allow the deformation of individual parts. Consequently, shapes that have puzzle-like structures (i.e., where each part has to fit into another) typically also produce visible artefacts. Furthermore, as we do not perform any functional analysis, some intermediate shapes do not look plausible.

# 6

# Data-driven Material Assignment


**Figure 6.1:** Given an input query 3D object without materials *(left)* our approach automatically assigns materials *(center)* using database information and suggests alternatives to the user *(insets)* which can be selected interactively to improve the automatic assignment *(right)*.

## 6.1 Introduction

Assigning materials to parts of a 3D object is a difficult and time consuming task that is performed by specially trained color & lighting artists in movie or game productions. The chosen palette of materials strongly influences the overall appearance of the 3D scene and is essential to allow the object to fit into an environment. There is a wide range of different materials, e. g., for a car there is the specular metallic paint work, the diffuse rubber material on the tire, the aluminium of the rim, the fabrics and leather used in the interior, etc. The compositing of materials is also important, as for example all screws on a tire should not just be metallic, but are likely to be of the same metallic material. Furthermore, materials are influenced by their context, e. g., for a part of a car's interior, leather or wood are far more likely materials than for a part of the car's engine.

Despite these observations, current content creation packages assign materials using a tedious manual process, involving the adjustment of rarely intuitive parameters, or by selecting pre-defined materials from a database using a keyword search. Even an

experienced artist requires approximately 45 minutes to assign appropriate materials to all 130 parts of the car shown in the Figure 6.1.

In this work, we propose an approach to computationally model the relation of shape and material by learning it from a database of hundreds of multi-component 3D objects with materials. This model can then be used to automatically assign materials to 3D objects or can be employed in a user-interface to provide a ranked list of the most likely materials (see Figure 6.1).

This chapter comprises the following contributions:

- A model of the relation between materials and shape as well as context, called the *material memex*,
- Automatic assignment of materials using this model.
- A novel interface to guide a user when assigning materials by providing ranked material suggestions.
- A user study of task performance when using conventional slider or text interfaces compared to our interface.

The rest of the chapter is structured as follows. The next section discusses related work. Section 6.2 introduces the proposed material memex. Section 6.3 presents two applications of the material memex: automatic material assignment and ranked material suggestion. Results are given in Section 6.4.

## 6.2 Material Memex

Input to our approach is a multi-component 3D object where parts have no assigned material. Output is a suggestion for a suitable material for each part. The part-material relation is learned from a database of multi-component 3D objects.

We follow the *memex model* recently popularized by Malisiewicz and Efros [Malisiewicz and Efros 2009], which is based on Vannevar Bush's [Bush 1945] early concept of a memory extender (memex). The memex does not structure information using categories, but rather stores associations between entities. As most entities have no strong pairwise associations, the memex is typically a sparse representation. Malisiewicz and Efros show that this category-free memex model outperforms category-based approaches on the challenge to detect an object in a 2D image using only contextual cues [Torralba 2003].

Similarly, we propose to store associations between parts and materials of database objects in a *material memex* to generate a suitable material suggestion for a user-provided query object. Using the sparse memex representation, the approach can efficiently compute a likelihood value for each possible material for a part of the

input query object. The computed likelihood value can be used, for example, to display a ranking of the best-fitting materials for a part of the query object to the user. The likelihood value depends on the shape similarity between parts, i. e., the probability for a candidate material is high if there are parts in the database that have a shape similar to the query part and a material similar to the candidate material. Furthermore, the likelihood value depends on the context of the part. In this work, the context information is captured by the pairwise spatial relation between two parts that are in physical contact. The probability for a candidate material is high if there are two parts in the database which are in physical contact and have similar materials and a similar spatial relationship as the query part and one of its contextual parts in the query object. Additionally, it is very likely that parts with similar shape in the query object have similar materials. For larger databases and query objects with many parts, the sparse memex representation, which can be built once during pre-processing, is the key ingredient making it possible to return material suggestions at interactive speed.

In contrast to previous work, in our application the context is not fully defined, i. e., in our case the probability for a candidate material depends on materials of contextual parts in the query objects, which are also unknown. Consequently, the problem addressed in this work has many unknown variables and the globally best solution must be estimated by maximizing the joint probability distribution of all variables simultaneously.

**Overview** In the following, we want to assemble a probabilistic factor graph that represents the joint probability distribution for all the unknown materials of the query object. The individual probability distributions for the material of each part are then inferred by the marginal distributions, which can be calculated with the sum-product algorithm (belief propagation).

The structure of the probabilistic factor graph depends on the query object and the potentials of the individual factors are dependent on the query object and all the objects in the database. In order to determine the structure of the factor graph, we need to determine which parts are in context and are consequently influencing each other. Here, we assume, for the sake of lower computational complexity, that parts only influence each other if they are in physical contact.

The following paragraphs will introduce the entities required to analyze the input data in order to create the factor graph. First, we describe what our approach expects as input, the decomposition of objects into parts, and the contact analysis. Afterwards, measures for the spatial relation between parts, for shape similarity, and for material similarity are introduced. The defined measures will allow us to create a sparse memex graph for the query and a sparse memex graph for the database objects. The query memex graph will directly lead to the structure of the probabilistic factor graph and the database memex graph is required to efficiently compute the potentials of the individual factors.

**Input.** Input to the approach is a multi-component 3D object $\hat{S}$ that has no assigned material, which is called the *query* object. Furthermore, there is a database of multi-component objects with assigned materials. This database is defined by the set of objects $\mathcal{S}$. This set should be large, e. g., in our experiments we used 276 objects. The individual database objects $S_m \in \mathcal{S}$ are given as polygonal meshes where each face is labeled with a material of the Phong [Phong 1975] reflection model. We use Phong in our implementation as it is a widespread reflection model for 3D objects from Internet repositories, such as Google 3D Warehouse, Turbosquid, Dosch 3D, etc. Uniform scaling is applied to all 3D objects in the database as well as the query shape. This ensures that all objects have the same size (measured on the dominant eigenvector of the covariance matrix of the mesh vertices). As we are not employing a category-based learning approach, neither the objects, polygons, or materials of the database require any manual annotations.

**Decomposition Into Parts.** The material memex stores associations between parts of objects. Before these associations can be computed, all objects must be decomposed into their parts, i. e., it must be defined which faces of an object's polygonal mesh belong to a particular part. The pre-processing pipeline as outlined in Section 2.2. For decomposition, we assume that the designers of the query and database objects have initially created the object from multiple parts. If this part information is still available, it is used for decomposition. However, for approximately 90 percent of our models the part information is lost, as it is typically not stored in the file formats utilized by the Internet repositories. In these cases, we recover the part information by searching for connected components in the polygonal mesh. For the database shapes it is also ensured that all polygons in the connected mesh component have the same material.

Let $\mathcal{P} := \{P_1, \ldots, P_k, \ldots, P_K\}$ be the set of parts that is created by the decomposition of all objects of the database into parts. As this set comprises all parts of the database, $K$ is the number of all parts in the complete database. Every part $P_k$ in the database has a unique material $M_k$. Note that in contrast to common practice in computer graphics, in the mathematical description used throughout this section, materials are not shared across different parts even if they are identical, i. e., every part has one material and every material has one part. Consequently, there is a set of known database materials $\mathcal{M} := \{M_1, \ldots, M_k, \ldots, M_K\}$ which has the same cardinality as the set of parts. The query object $\hat{S}$ is decomposed in exactly the same way, resulting in a set of parts $\hat{\mathcal{P}} := \{\hat{P}_1, \ldots, \hat{P}_n, \ldots, \hat{P}_N\}$.

**Contact Analysis** For the query object $\hat{S}$ and each database object $S_m \in \mathcal{S}$ a contact analysis is performed. The contact analysis of the query object will be later used to generate the query memex graph (which in turn captures the structure of the probabilistic factor graph). The contact analysis of the database objects is performed to compare the spatial context of a part in the query object to a spatial context of a

part in the database, as will be explained in detail later.

During contact analysis we compute whether the two parts, $P_i$ and $P_j$, of an object are in physical contact, i. e., are (almost) touching each other. To this end, both parts are resampled to a point cloud. The individual sampling points are placed on the polygonal surface mesh such that their distance is roughly equal (blue noise sampling). We define the minimal spatial distance $d_{i,j}^{\min}$ as the smallest Euclidean distance between any sampled point from part $P_i$ to any sampled point from part $P_j$. We define two parts $P_i$ and $P_j$ to be in contact if their minimal spatial distance $d_{i,j}^{\min}$ is smaller than 0.01 percent of the total object size.

In the following, we describe three associations between parts that are required to build the material memex: *spatial relation* similarity for part pairings, *shape* similarity, and *material* similarity.

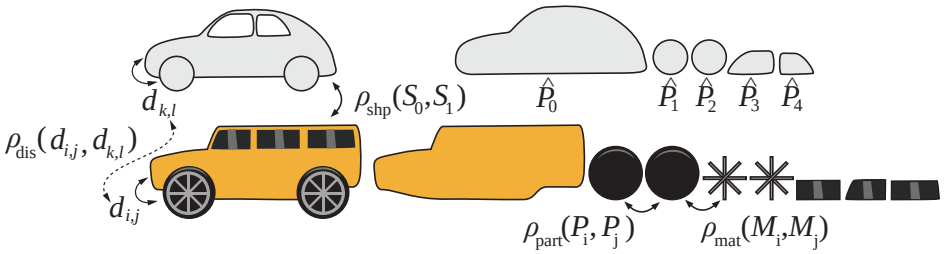



**Figure 6.2:** Prerequisites to build a material memex graph. The query object *(top)* and a database object *(bottom)* are decomposed into parts. We defined several associations between parts: spatial relation similarity $\rho_{\text{dis}}$, shape similarity $\rho_{\text{part}}$ and $\rho_{\text{shp}}$, and material similarity $\rho_{\text{mat}}$.

**Spatial Relationship of Parts** Besides the structural information produced by the contact analysis, spatial relation similarity for part pairings is another important cue to determine whether a context in the query object and in the database object is similar. The Euclidean distances $d_{i,j}$ between the centers of mass of parts $P_i$ and $P_j$ is used to capture their spatial relationship.

In our approach, we have made the design decision to rely solely on Euclidean distances between parts to capture their spatial relationship.

The advantage of using the Euclidean distances between parts instead of a more complex spatial relation measure is that the Euclidean distance is invariant to joint rotational and translational transformation of the two involved parts. Consequently, the measure is not dependent on the choice of the local coordinate system, which is typically not consistent for the parts of the 3D objects. However, the measure is not scale-invariant, which is why we initially scale all objects to have the same size.

To define a similarity measure of two spatial relations $d_{i,j}$ and $d_{k,l}$, it is assumed that due to noise the difference between two spatial relations (that should be considered similar) obeys a Gaussian distribution with zero mean. This is motivated by the central limit theorem. We define the similarity measure of two spatial relations $d_{i,j}$

and $d_{k,l}$ as

$$\rho_{\text{dis}}(d_{i,j}, d_{k,l}) = \exp\left(\frac{-(d_{i,j} - d_{k,l})^2}{2\sigma_{\text{d}}^2}\right) \quad , \tag{6.1}$$

where $\sigma_{\text{d}}$ denotes the standard deviation of spatial relations difference $(d_{i,j} - d_{k,l})$ that should be considered similar. We chose $\sigma_{\text{d}}$ such that the resulting zero-mean Gaussian distribution captures the smallest $\tau_{\sigma_{\text{d}}} = 3.0$ percent of all spatial relation differences in the complete database of parts. This spatial similarity measure is close to 1.0 if the two spatial relations are very similar and close to 0.0 if they are very different.

**Shape Similarity.** The similarity $\rho_{\text{shp}}(P_i, P_j)$ between the parts $P_i$ and $P_j$ is calculated using the approach by Chen et al. [Chen et al. 2003]. In our application, the scale invariance of their descriptor is undesirable. We extend it by adding three entries to their descriptor vector that represent each shape's size in the $x$-, $y$- and $z$-directions measured in the space spanned by the three eigenvectors of the covariance matrix of the part's sampling points.

We measure the dissimilarity between part $P_i$ and part $P_j$ using the L1-distance $|\mathbf{s}(P_i) - \mathbf{s}(P_j)|$ of the extended descriptor vectors. Assuming a Gaussian distribution of the L1-distances that should be considered similar, we define the shape similarity of parts to be

$$\rho_{\text{part}}(P_i, P_j) = \exp\left(\frac{-|\mathbf{s}(P_i) - \mathbf{s}(P_j)|^2}{\sigma_{\text{p}}(P_i, P_j)^2}\right) \quad . \tag{6.2}$$

We chose the standard deviation $\sigma_{\text{p}}(P_i, P_j)$ such that the resulting zero-mean Gaussian distribution captures the smallest $\tau_{\sigma_{\text{p}}} = 1.0$ percent of all L1-distances that involve either $P_i$ or $P_j$ in the complete database. Due to this definition, the standard deviation $\sigma_{\text{p}}(P_i, P_j)$ is a function of $P_i$ or $P_j$. This is required, as we have observed in our experiments that the absolute value of the L1 distance depends strongly on the involved parts $P_i$ or $P_j$. This shape similarity measure is close to 1.0 if the two parts are very similar and close to 0.0 if they are very different.

Besides geometric shape similarity between parts, shape similarity can also be computed between whole objects. In particular, we will require a measure for the shape similarity $\rho_{\text{shp}}(\hat{S}, S_m)$ between the query object $\hat{S}$ and an object in the database $S_m$, when computing the potentials of the factor graph. When computing these potentials we can weight the influence of a part higher if it belongs to an object that is similar to the query object.

To compute $\rho_{\text{shp}}(\hat{S}, S_m)$, we again employ the approach of Chen et al. but now rendering and comparing whole objects and not parts. The only difference is that we extend the aligned descriptor created by Chen et al.'s approach differently. When

comparing objects we only add a single entry, which is the number of parts of the object. This is motivated by the fact that our approach works best if the query and database objects have compatible complexity because the part segmentation is more likely to be similar in this case. Adding the number of parts in the shape similarity object has the result that parts from objects with similar complexity are preferred when computing the potentials of the factor graph.

Formally, we define the shape similarity of objects to be

$$\rho_{\text{shp}}(\hat{S}, S_m) = \exp\left(\frac{-|\mathbf{s}(\hat{S}) - \mathbf{s}(S_m)|^2}{\sigma_{\text{s}}(\hat{S}, S_m)^2}\right) \quad . \tag{6.3}$$

We choose the standard deviation $\sigma_{\text{s}}(\hat{S}, S_m)$ such that the resulting zero-mean Gaussian distribution would capture the smallest 15 L1-distances between $\hat{S}$ and all database shapes $S_m \in \mathcal{S}$.

**Material Similarity.** The material similarity function between material $M_i$ and $M_j$ is denoted by $\rho_{\text{mat}}(M_i, M_j)$. We employ a custom difference of Phong reflection parameters inspired by the perceptual re-parameterization by Pellacini et al. [Pellacini, Ferwerda and Greenberg 2000]. In particular, the specular color is parameterized by the third root of RGB color values instead of using the RGB values directly. Further, we have a parameter for the glossiness of the material (where we define glossiness to be exponent of the cosine function typically used to compute the specular component). We divide Phong glossiness by an assumed maximum of 300 and afterwards take the forth root to linearize its perceptual influence. Consequently, our Phong material descriptor vector contains eight elements: 3 for the diffuse color values in RGB color space, 3 for the third root of specular color values in RGB color space, 1 for the fourth root of the glossiness, and 1 for transparency. All descriptor elements can take values in the range of 0.0 to 1.0.

The L1-distance $|\mathbf{m}(M_i) - \mathbf{m}(M_j)|$ between the material descriptor vectors $\mathbf{m}(M_i)$ and $\mathbf{m}(M_j)$ is used to measure the dissimilarity between material $M_i$ and part $M_j$. Assuming again a Gaussian distribution of the L1-distances, the material similarity function is

$$\rho_{\text{mat}}(P_i, P_j) = \exp\left(\frac{-|\mathbf{m}(M_i) - \mathbf{m}(M_j)|^2}{\sigma_{\text{m}}^2}\right) \quad , \tag{6.4}$$

where $\sigma_{\text{m}}$ denotes the standard deviation over all L1-distances of materials in the complete database that should be considered similar. We chose $\sigma_{\text{m}}$ such that it captures the smallest $\tau_{\sigma_{\text{m}}} = 1.0$ percent of all L1-distances in the complete database.

**Database Memex Graph.** For all parts in the database we extract a Memex graph $\mathcal{G} = (\mathcal{V}, \mathcal{E}_{\text{d}} \cup \mathcal{E}_{\text{s}} \cup \mathcal{E}_{\text{m}})$ consisting of a set of nodes $\mathcal{V}$ and the union of the three sets of edges $\mathcal{E}_{\text{d}}$, $\mathcal{E}_{\text{s}}$, and $\mathcal{E}_{\text{m}}$. The set of nodes $\mathcal{V}$ is equivalent to the set of parts $\mathcal{P}$, i. e.,

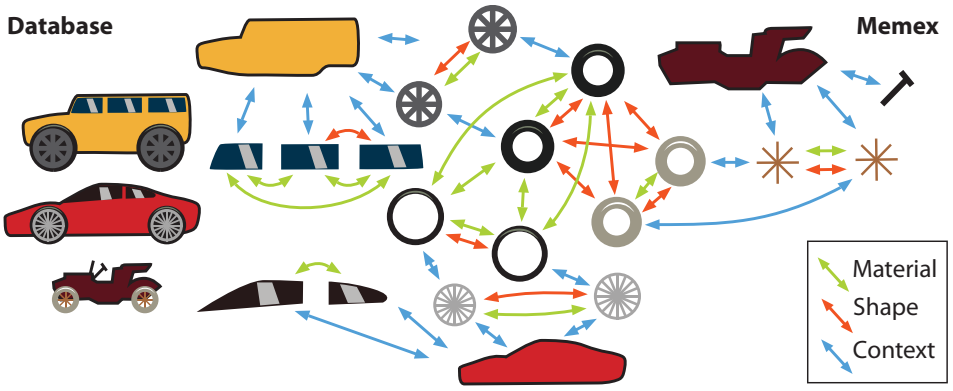



**Figure 6.3:** A database of objects *(top)* and the database memex graph that we extract *(bottom)* with spatial context, shape similarly, and material similarity edges.

each node represents a part in the database (Figure 6.3). The set of spatial context edges $\mathcal{E}_\mathrm{d}$ is determined by finding parts that are in contact (as defined during the contact analysis). If two parts $P_i$ and $P_j$, are in contact, their index pair $(i, j)$ is added to the set of edges $\mathcal{E}_\mathrm{d}$. Note, that only parts from the same object in the database can be in contact. This results in very sparse connectivity of nodes. In fact, the cardinality of $\mathcal{E}_\mathrm{d}$ grows only linearly with the number of objects in the database. Additionally, the graph contains a set of shape similarity edges $\mathcal{E}_\mathrm{s}$. If two parts $P_i$ and $P_j$, are of similar shape, i.e., their shape similarity function $\rho_\mathrm{part}(P_i, P_j)$ is above a user defined threshold, their index pair $(i, j)$ is added to the set of edges $\mathcal{E}_\mathrm{s}$. The third type of edges are material similarity edges that are added to the set $\mathcal{E}_\mathrm{m}$ if the material similarity $\rho_\mathrm{mat}(M_i, M_j)$ is above a user-defined threshold. Table 6.1 lists all symbols used in the text.

**Query Memex Graph.** Similarly, a graph $\hat{\mathcal{G}} = (\hat{\mathcal{V}}, \hat{\mathcal{E}}_\mathrm{d} \cup \hat{\mathcal{E}}_\mathrm{s})$ for the query object $\hat{S}$ is generated. The set of nodes $\hat{\mathcal{V}}$ is defined by the set of parts $\hat{\mathcal{P}}$ in the query object. The set of spatial context edges $\hat{\mathcal{E}}_\mathrm{d}$ and the set of part similarity edges $\hat{\mathcal{E}}_\mathrm{s}$ is generated in exactly the same way as for the database objects. Obviously, in contrast to the database memex graph, the query memex graph has no material similarity edges because the materials of the query object are unknown. In contrast to the database memex graph, the query memex graph cannot be computed in a pre-processing step, but rather only once the query object is defined.

**Probabilistic Factor Graph.** Using the information stored in the query and database memex graphs, we can generate a factor graph for probabilistic inference to find an optimal material assignment for each part of the query object. The structure of the factor graph is directly given by the query memex graph $\hat{\mathcal{G}} = (\hat{\mathcal{V}}, \hat{\mathcal{E}}_\mathrm{d} \cup \hat{\mathcal{E}}_\mathrm{s})$ as described in the following. For each node $\hat{\mathcal{V}}$ in the query memex graph a random element vertex in the factor graph is created. This random element represents the

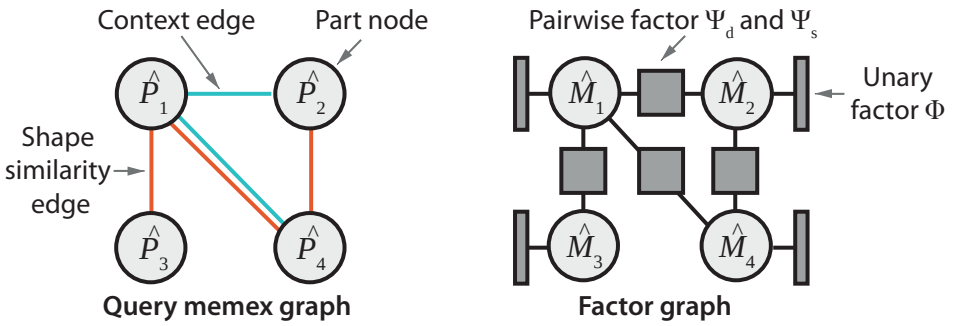



**Figure 6.4:** Converting the memex graph of the query object into a factor graph for probabilistic inference.

unknown material $\hat{M}_n$ of part $\hat{P}_n$ (see Figure 6.4). In total, $N$ random element vertices are created. Furthermore, the factor graph contains unary and pairwise factors.

For each unknown material $\hat{M}_n$, a unary factor is created and linked to the corresponding random element vertex in the factor graph. We define the unary potential of these factors to be

$$\phi(\hat{M}_n) = \sum_{k=1}^{K} \rho_{\text{mat}}(\hat{M}_n, M_k)\, \rho_{\text{part}}(\hat{P}_n, P_k)\, \rho_{\text{shp}}(\hat{S}, S(P_k)) \quad , \tag{6.5}$$

which depends on all parts $P_k$ and their materials $M_k$ in the database. Our choice for the unary potential in Eq. 6.5 is motivated by the intuitive assumption that the potential should be large if there is a part $P_k$ in the database that has a similar material as $\hat{M}_n$ and similar shape as $\hat{P}_n$. Furthermore, the contribution of a part $P_k$ is weighted by the shape similarity between the query object and the object $S(P_k)$ containing part $P_k$.

For each graph edge of the query memex graph in both sets $\hat{\mathcal{E}}_{\text{d}}$ and $\hat{\mathcal{E}}_{\text{s}}$ a pairwise factor is created. This pairwise factor is linked to the corresponding vertices in the factor graph, as illustrated in Figure 6.4.

If $(i, j) \in \hat{\mathcal{E}}_{\text{d}}$, which means that it is a spatial context edge, the pairwise potential of the factor is defined by

$$\begin{aligned} \psi_{\text{d}}(\hat{M}_i, \hat{M}_j) = \frac{1}{|\mathcal{E}_{\text{d}}|} \sum_{(u,v)\in\mathcal{E}_{\text{d}}} \big[ & \rho_{\text{mat}}(\hat{M}_i, M_u)\rho_{\text{mat}}(\hat{M}_j, M_v) \\ & \rho_{\text{part}}(\hat{P}_i, P_u)\rho_{\text{part}}(\hat{P}_j, P_v) \\ & \rho_{\text{dis}}(d_{i,j}, d_{u,v})\, \rho_{\text{shp}}(\hat{S}, S(P_u))\big] \,. \end{aligned} \tag{6.6}$$

Otherwise, if $(i, j) \in \hat{\mathcal{E}}_s$, which means that it is a shape similarity edge, the pairwise potential is simply

$$\psi_{\text{s}}(\hat{M}_i, \hat{M}_j) = \rho_{\text{mat}}(\hat{M}_i, \hat{M}_j) \quad . \tag{6.7}$$

**Table 6.1:** Table of symbols.

| Entity | Symbol |
|---|---|
| Set of all polygonal meshes | $\mathbb{S}$ |
| Set of all materials | $\mathbb{M}$ |
| Database object | $S \in \mathbb{S}$ |
| Number objects | $L \in \mathbb{N}$ |
| Set of database objects | $\mathcal{S} \in \mathbb{S}^L$ |
| Number database parts | $K \in \mathbb{N}$ |
| Set of database parts | $\mathcal{P} := \{P_1, \ldots, P_k, \ldots, P_K\} \in \mathbb{S}^K$ |
| Set of database materials | $\mathcal{M} := \{M_1, \ldots, M_k, \ldots, M_K\} \in \mathbb{M}^K$ |
| Query object | $\hat{S} \in \mathbb{S}$ |
| Number of query parts | $N \in \mathbb{N}$ |
| Set of query parts | $\hat{\mathcal{P}} := \{\hat{P}_1, \ldots, \hat{P}_n, \ldots, \hat{P}_N\} \in \mathbb{S}^N$ |
| Spatial distance (context) | $d_{i,j} \in \mathbb{N}^2 \rightarrow \mathbb{R}$ |
| Spatial context similarity | $\rho_{\text{dis}}(d_{i,j}, d_{k,l})$ |
| Shape similarity | $\rho_{\text{shp}}(P_i, P_j) : \mathbb{N}^2 \rightarrow \mathbb{R}$ |
| Material similarity | $\rho_{\text{mat}}(\hat{M}_i, M_u) : \mathbb{N}^2 \rightarrow \mathbb{R}$ |
| Database memex graph | $\mathcal{G} = (\mathcal{V}, \mathcal{E}_d, \mathcal{E}_s, \mathcal{E}_m)$ |
| Query memex graph | $\hat{\mathcal{G}} = (\hat{\mathcal{V}}, \hat{\mathcal{E}}_d, \hat{\mathcal{E}}_s)$ |
| Unary potential | $\phi(\hat{M}_n) : \mathbb{M} \rightarrow \mathbb{R}$ |
| Pairwise potential | $\psi(\hat{M}_i, \hat{M}_j) : \mathbb{M} \rightarrow \mathbb{R}$ |
| Helper constants | $\alpha, \beta, \gamma \in \mathbb{R}$ |

The pairwise potential in Eq. 6.6 for spatial context edges is high if the two parts of the query object $\hat{P}_i$ and $\hat{P}_j$ have similar materials, similar shape, and a similar spatial relation as two parts $P_u$ and $P_v$ from the database. The potential is weighted by the shape similarly between the query object and the object $S(P_u)$ containing part $P_u$ and $P_v$.

The pairwise potential in Eq. 6.7 for shape similarity edges is high if parts with similar shape in the query object have similar material.

As the number of parts and the cardinality of $\mathcal{E}_d$ grows linearly with the number of objects in the database and only the database memex can be precomputed, the computation time for the factors at runtime also grows linearly. However, by defining a threshold on shape similarity $\rho_{\text{shp}}(\hat{S}, S(P_u))$ between the query and the database objects in Eqs. 6.5 and 6.6 a constant computation time can be achieved once the shape similarity is known. By applying the threshold, the summation includes only those parts and edges that belong to similar database objects, which is an approximately constant quantity (due to our choice of the shape similarity of objects in Eq. 6.3).

**Inference** The factor graph defined in the previous paragraph represents the joint probability distribution of all the unknown materials $\hat{M}_n \in \hat{\mathcal{M}}$ of the query object, which is given by

$$P(\hat{\mathcal{M}}) = \frac{1}{Z} \prod_{\hat{M}_n \in \hat{\mathcal{M}}} \phi(\hat{M}_n) \prod_{(i,j) \in \hat{\mathcal{E}}_\mathrm{d}} \psi_\mathrm{d}(\hat{M}_i, \hat{M}_j)^\alpha \prod_{(i,j) \in \hat{\mathcal{E}}_\mathrm{s}} \psi_\mathrm{s}(\hat{M}_i, \hat{M}_j)^\beta, \qquad (6.8)$$

where $Z$ is a normalizing constant known as the *partition function*. The exponents $\alpha = 1.0$ and $\beta = 20.0$ are weighting parameters. Note, that these exponents correspond to linear weighting multipliers of the pairwise terms if the joint probability distribution is transferred in the log-domain as is typically done during inference.

In theory, the unknown materials $\hat{M}_n$ are continuous entities. However, for efficient inference using the factor graph, we define a discrete set of candidate materials $\mathcal{C}$. This set is derived from the set $\mathcal{M}$ of all the materials in the database. We cluster the database materials using a maximum of $D = 100$ cluster centers. Afterwards, for each cluster center, the database material $M_n \in \mathcal{M}$ with the smallest distance $\rho_\mathrm{mat}$ to the cluster center is added to the set $\mathcal{C}$ of candidate materials. Consequently, the pairwise factors of the factor graph can be represented as $|\mathcal{C}| \times |\mathcal{C}|$ matrices and the unary factors are $|\mathcal{C}| \times 1$ vectors (see Fig 6.4).

The task is now to assign a candidate material from set $\mathcal{C}$ to each random variable of the factor graph, i. e., to each material $\hat{M}_n \in \hat{\mathcal{M}}$, so that the marginals $P(\hat{M}_n)$ of the joint probability $P(\hat{\mathcal{M}})$ are maximized:

$$\arg\max_{\hat{M}_n} P(\hat{M}_n) = \arg\max_{\hat{M}_n} \left( \sum_{\hat{\mathcal{M}} \setminus \hat{M}_n} P(\hat{\mathcal{M}}) \right) \quad \forall \quad \hat{M}_n \in \hat{\mathcal{M}} \qquad (6.9)$$

These maxima can be approximated efficiently using the sum-product algorithm (loopy belief propagation) on the factor graph [Kschischang, Frey and Loeliger 1998]. The required computation time can be further reduced by using a GPU shader implementation with parallel message updating. Our CPU implementation requires 4897 ms for 50 iterations on a factor graph with 211 unary factors and 623 pairwise factors for 50 candidate materials. The parallel GPU implementation requires 158 ms for the same task on an NVIDIA Geforce GTX 275.

## 6.3 Applications

In the following, we describe different applications that become possible with the material memex model, such as automatic assignment of materials to a 3D object or providing user assistance when assigning materials.

### 6.3.1 Automatic Material Assignment

To automatically assign materials to a query shape without materials, Eq. 6.9, i. e., the maximum marginal for each random variables $\hat{M}_n$, needs to be solved, resulting in a set of materials that can be interactively previewed. The entire material memex is pre-computed and serialized to disk (10 hours for a database of 276 shapes). The most time-consuming steps at run-time are computing the unary (Eq. 6.5) and the pairwise factors (Eqs. 6.6 and 6.7) as well as executing the loopy belief propagation. Fortunately, both steps can be performed using a GPU: dependent on the size of the query object, computing factors takes 500 to 3000 ms, and resp. 20 ms to 200 ms for the loopy belief propagation. Please note, that pre-computation of the factors is not possible as they depend on the choice of the query object.

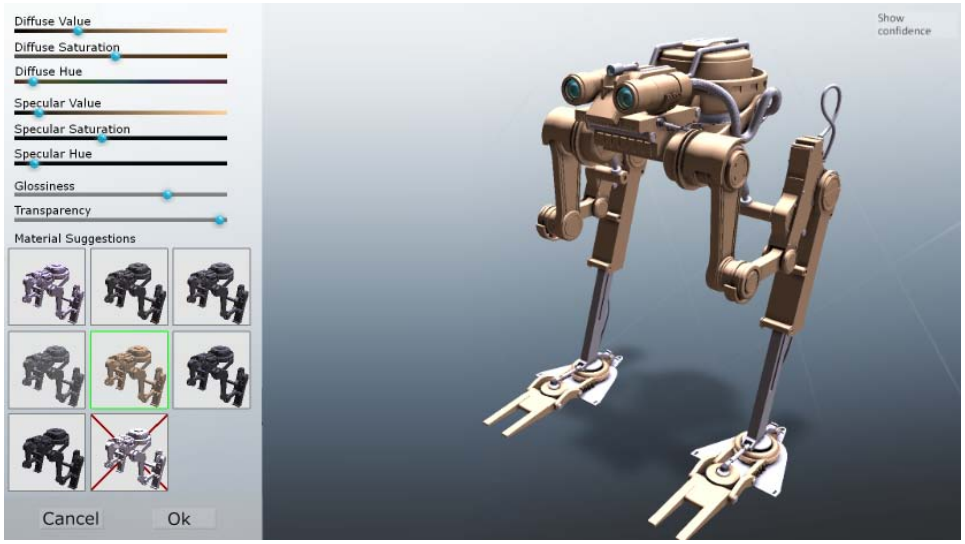



**Figure 6.5:** The user interface for ranked material suggestion.


**Figure 6.6:** Automatic material assignment: database of cars where the materials are assigned by a human designer *(1st row)*, results of automatic material assignment *(2nd row)*. Results are generated by leaving out the particular query car from the database.

### 6.3.2 Ranked Material Suggestion

Besides assigning the top material to each part of a query shape, our approach can also be used to present an ordered list of alternative material assignments. In our prototype (see Figure 6.5), after a fully-automatic initial assignment, a user can freely select parts, and is presented a list of alternative material suggestions, ranked by likelihood (Eq. 6.9). After selecting a suggestion as a part's material, this material becomes *fixated*, i. e., its random variable is removed from the factor graph, and all neighboring pairwise factors become unary factors by keeping only the row (or column) of the pairwise factor matrix that represented the fixated material. After fixation, the inference is restarted, possibly changing other materials in the query, e. g., after assigning a black rubber material to a tire of a car, all other tires are very likely to be assigned the same material if they are linked with a part similarity edge. To remove a shape's material fixation, the random variable is re-included in the factor graph and the unary and pairwise factors are restored.

Furthermore, the user has the option to create a new material for a selected part using a slider interface to control the parameters of the perceptually parameterized Phong [Kerr and Pellacini 2010] reflectance model (see Figure 6.5). New materials are added to the set $\mathcal{C}$ of candidate materials. Consequently, all unary factor vectors get an additional element, and all pairwise factor matrices get an additional row and column. The user can seamlessly repeat this procedure as many times as required to achieve the desired goal.

## 6.4 Results

In this section, several results that are generated with the proposed approach are presented. Similar results as well as a demonstration of our interactive user-interface for material suggestion can be seen in video[1] and further details can be found on the project web-site[2]. All results shown in this section are generated based on the same database that contains 276 objects with known material that we collected from internet repositories.

Figure 6.6 shows results of the fully automatic assignment of materials. Six different car models are employed as query objects and the automatic material assignment is run six times producing the six results shown in the 2nd row of the figure. The car models have a complexity of 120 to 250 parts. Figure 6.6 also shows the manual material assignment created by a designer. Creating such a manual assignment for a single car takes approximately 45 minutes.

It can be observed that the difference in quality between the designer and automatic material assignment is small. Almost all of the resulting automatic material assign-

[1] `https://www.youtube.com/watch?v=ekDoZMyP4L4`
[2] `http://resources.mpi-inf.mpg.de/MaterialMemex/`

ments look very plausible, e.g., all cars have specular body colors and diffuse dark rubber material at the tires; the windows are transparent; also details, such as the disk brake calipers, indicators, or head lights, have suitable material assignments. However, there are also several wrong material assignments, e.g., the hood cover of the convertible is transparent. Such estimation errors are more likely to occur if parts with similar shapes and spatial configurations are not observed in the database. This is also illustrated in Figure 6.7, where the query object is an unusual aircraft and the most similar objects in the database are of quite different objects. As there are no suitable pairwise associations in the database, the material assignment does not look realistic.

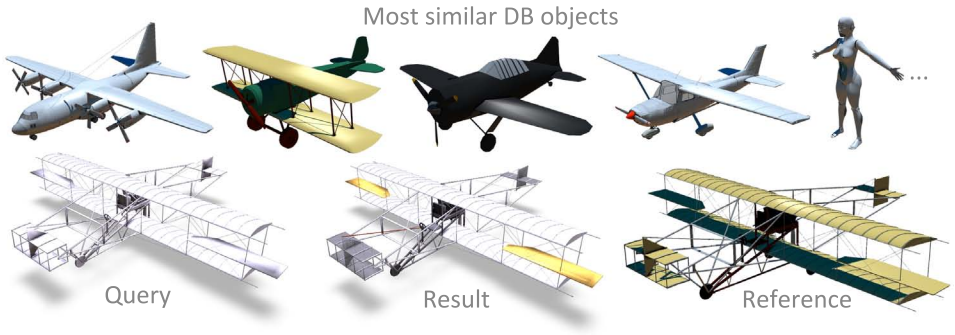



**Figure 6.7:** Failure case for automatic material assignment: database *(top)*, query object *(left)*, our result *(middle)*, reference *(right)*.


**Figure 6.8:** Comparison of using only the unary factors during inference *(left)* vs. unary and pairwise factors *(right)*.

Figure 6.8 is a comparison of a result where only the unary factors (see Eq. 6.5) are employed for an automatic material assignment, versus our approach of using unary factors and pairwise factors in a probabilistic factor graph. In the result that uses only the unary term, several wrong assignments are visible. Using context information that is stored in the pairwise factors allows removing these artefacts.

Figure 6.9 shows several examples where the proposed user-interface for ranked material suggestion is employed to refine an automatic assignment. All shown refinements required a maximum of two manual material selections from the ranked

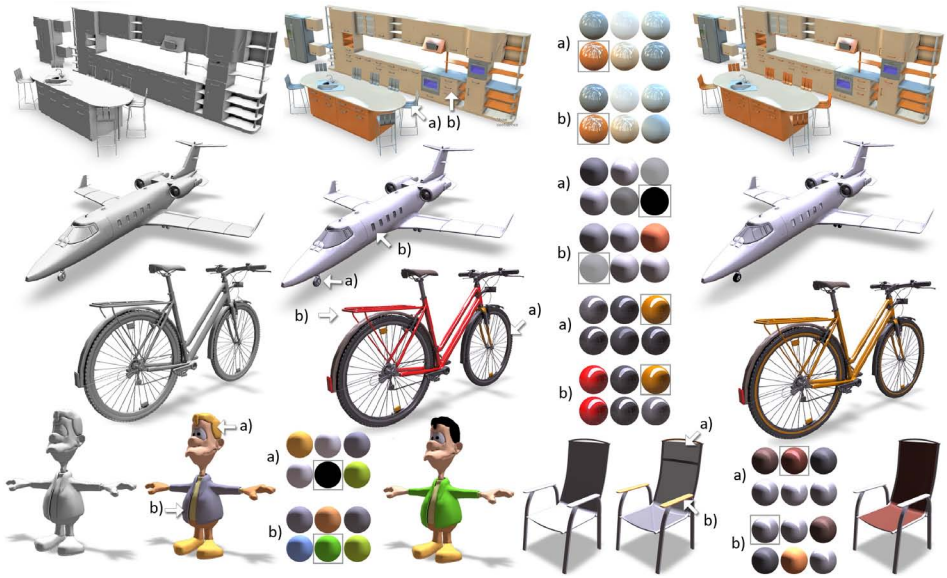



**Figure 6.9:** Material suggestion *(left to right)*: query object without material, results of automatic material assignment, user interaction (arrows indicate the clicked part , colored spheres show suggested materials ranked by likelihood), refined results after the user interaction.

list to achieve the shown results. It is evident that the approach works for a large range of query objects with different complexity and topology. A user can employ our interface to generate similar results. It also visualizes those database objects, which are found to be most similar to the individual query objects.

Furthermore, shape similarity edges and spatial context edges can cause other parts to change their automatic material assignment when a user manually assigns a material to a part. The spatial context edges typically influence only other parts in the vicinity of the part. It can be observed that for query objects with a smaller factor graph, material changes occur more often. For very complex objects, where most random elements of the factor graph have many connected edges, manual assignments of a material typically do not influence the margins of the other parts strongly enough to provoke a material change.

Another application of our approach is shown in Figure 6.10. If a single object is added to the database and the approach is run on several query objects, all query objects will be given similar material assignments. This can be employed, e. g., to indicate that the resulting group belongs to the same team in a computer game.

**User study.** The effectiveness of our approach is evaluated in a user study. The first study shows that our interface significantly improves human task performance in a material-assignment session relative to other common user interfaces. The second study shows, that the suggested approach produces better material assignments than common user interfaces. Third, we show how simpler methods for material

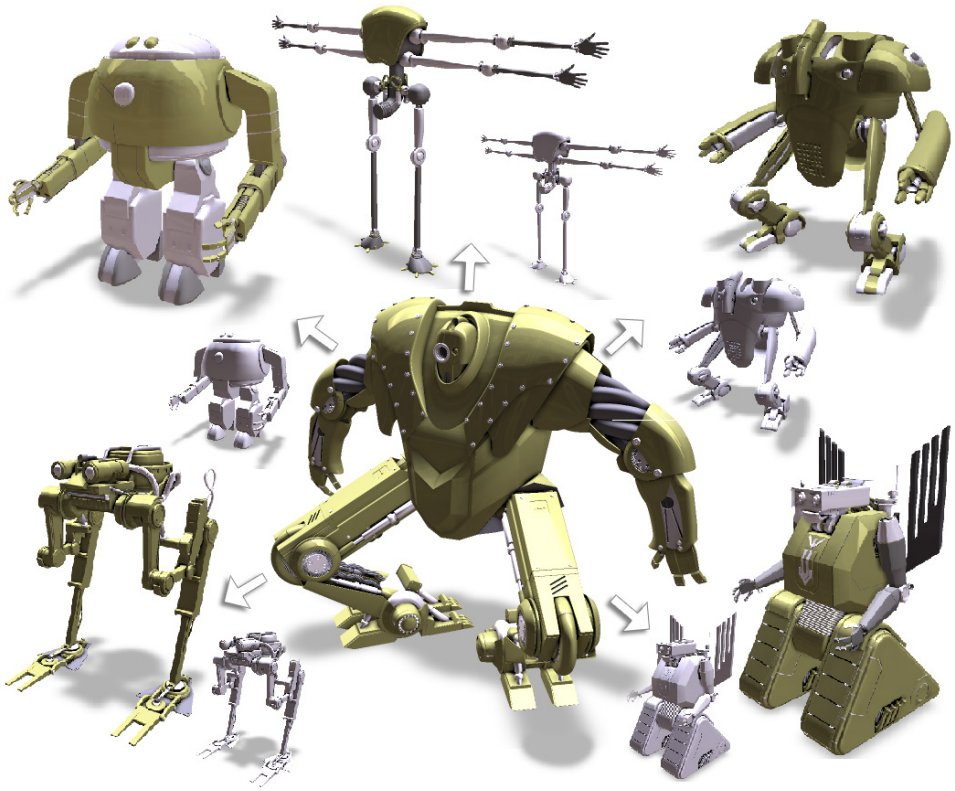


**Figure 6.10:** Material transfer from a single database object to multiple other objects: database object *(center)*, query objects *(inner circle)*, query object with transferred materials *(outer circle)*.

assignment will likely fail to achieve the same quality as our approach. Finally, a study of material assignment preference over database size indicates that our database sizes are sufficient.

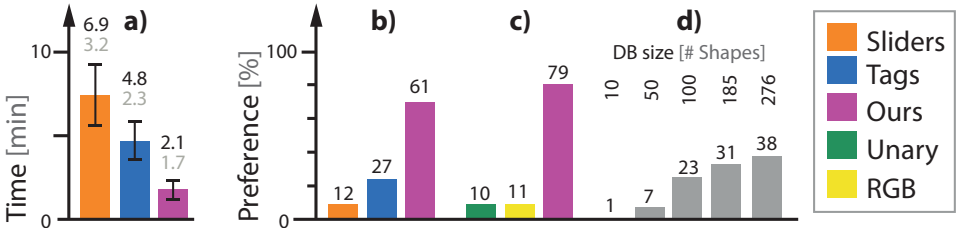



**Figure 6.11:** Results of our study (Please see text).

First, we compare task performance in terms of completion time achieved by users of our interface to two other common user interfaces. The first interface uses perceptual *sliders* and color pickers to select the parameters of the Phong reflection model. In the second interface, a user specifies one or multiple *keywords* by typing, and the interface presents preview icons of all materials that partially match one or multiple given keyword. The user selects the final material by clicking an icon. The database of 86 materials was labeled manually with keywords. Note that such keywords are typically not available. In total, 24 subjects participated in the first study. After a short tutorial, each subject performed 6 trial experiments. In each, the participant was asked to assign materials to two objects using three interfaces (sliders, keywords, or ours) in a random order. The maximum time allowed to complete the task was 10 minutes. A significant difference was found between all interfaces and an improvement of 227 s ($p < 10^{-15}$) comparing ours to the slider and 105 s ($p < 10^{-8}$) when comparing ours to the keyword interface (Figure 6.11a). This suggests that our interface can save considerable time and effort.

In a second study, 57 subjects were presented 10 random pairs of images produced with the different approaches in the previous study and had to choose the one with the better material assignment in a two-answer forced choice (2AFC) task. Results from our method were chosen to be preferred in 61 percent of all answers. Our method is significantly better when comparing its score to sliders ($p < 10^{-15}$) and keyword tags ($p < 10^{-9}$), see Figure 6.11b.

In a third experiment, we compare our approach against two simpler alternatives: a hypothetical method without a graphical model that only uses unary potentials but no pairwise potentials and a method that does not account for materials and uses only the diffuse RGB color during inference. In a 2AFC task (20 subjects, 10 trials each), we find a significant difference between our approach and simpler alternatives (Figure 6.11c), suggesting that material assignment is context-dependent ($p < 10^{-7}$), and a suitable representation of materials is crucial for plausible material assignment ($p < 10^{-8}$).

The last experiment investigates the dependency of quality on database size (Fig-

ure 6.11d). Subjects were presented images of objects with our material assignment, but produced from databases of increasing size. Again, subjects were asked to select the preferred material composition when presented 320 random pairs from all four database sizes in a 2AFC (details are given as supplemental material[3]). While there is a significant difference between databases with 10 ($p < 10^{-15}$), 50 ($p < 10^{-9}$), 100 ($p = 0.0063$) shapes, no significant difference is found between 185 and 276 shapes ($p = 0.042$). This indicates, but does not prove, that there is only a small effect when increasing the database size beyond 185 exemplars, i. e., that the database size used gives a good indication of the achievable quality.

[3] http://resources.mpi-inf.mpg.de/MaterialMemex/userstudy_results.pdf

# 7 Conclusion

In this thesis we discussed three main tracks of contribution: Data-driven Video Editing, Data-driven 3D Modeling, and Data-driven Material Assignment. Concluding remarks on the thesis are given in Section 7.1, with an outlook on possible future work in Section 7.2.

## 7.1 Closing Remarks

In this section we give some concluding remarks on the contributions made in chapters Chapter 4 through Chapter 6.

### 7.1.1 Data-driven Video Editing

Chapter 4 introduced *MovieReshape*, a system for performing realistic spatio-temporal reshaping of human actors in video sequences. Our approach is based on a statistical model of human shape and pose which is tracked to follow the motion of the actor. Spatio-temporally coherent shape edits can be performed efficiently by simply modifying a set of semantically meaningful shape attributes. We have demonstrated the high visual quality of our results on a variety of video sequences of different formats and origins, and validated our approach in a user study. Our system paves the way for previously unseen post-processing applications in movie and video productions.

### 7.1.2 Data-driven 3D Modeling

In Chapter 5 we addressed the problem of interactively composing complex shapes from individual parts such as those found in our everyday environment. We exploit

the constraints of symmetries and physical contacts, as analyzed from the example shapes, to restrict the produced shapes. As a result, visually pleasing blends can be rapidly generated between a large number of database shapes of different structures. Combining our approach with a more explicit functional analysis would be a challenging but interesting extension of our system that may improve the plausibility of the results.

### 7.1.3 Data-driven Material Editing

Chapter 6 proposed an approach to model the relation of shape and material computationally by learning it from a database of 3D objects with materials. It represents associations between database parts, i. e., whether two parts have similar shape, similar material, or if they are in context. This information allows for automatically assigning materials to other 3D objects and assists users when designing materials by providing ranked material suggestions.

## 7.2 Future Work

In this section, possible avenues of future research are discussed.

There are severe limitations in the ability to create detailed visual content, be it images, videos or 3D models. It remains a difficult task accessible only to a niche group of trained users, with complex interaction required at several stages. Consequently, content production remains rather costly and poses a potential barrier to many applications.

Growth in the capabilities of consumer grade hardware such as GPU based computers, multi-core cell processor based gaming devices (such as Microsoft XBox® and Sony PlayStation®), depth sensors (e.g. Microsoft Kinect®) and smart mobile phones empower the average home user to be able to create content in the form of images, pictures or 3D models [Newcombe et al. 2011], [Snavely, Seitz and Szeliski 2006]. The user then also has the possibility to consume and share this content in a variety of ways. However, tools to process and manipulate this data acquired by the user remain tedious and cumbersome to use.

In this thesis we showed how data-driven approaches can lead to better tools for visual content creation and manipulation. We concur with the leading data analyst Anand Rajaraman's hypothesis that "*adding more, independent data usually beats out designing ever-better algorithms to analyze an existing data set*", and predict that data-driven techniques will become increasingly popular for designing the next generation of content creation tools.

### 7.2.1 Data-driven Video Manipulation

In Chapter 4 we showed that our approach can modify the body shape of actors in videos realistically. But if the pose tracking was sub-optimal, deformation constraints may be placed very close to or in the scene background. In this case, the image deformation applied to the actor may propagate into the background, leading to a halo-like warp. When the person's shape is extremely enlarged, distortions may become noticeable in the background (Figure 4.13). Similarly, when the person's apparent size is strongly reduced, the background is warped to fill the whole, whereas another option would be a spatio-temporal inpainting of the disocclusions. In the future, we plan to include inpainting functionality and apply a more advanced contour tracking and automatic segmentation approach. Another problematic situation arises when limbs are occluding other parts of the body. In this case the deformation of the occluded body part is also applied to the limbs, which is an undesired artifact. To this end, we would like to explore the possibility of using a layered 2.5D mesh, as [Hornung, Dekkers and Kobbelt 2007].

While our system works for people dressed in normal apparel, our approach might face difficulties when people wear very loose clothing, such as a full skirt or long coat. In such cases, automatic pose tracking would fail. In addition, our warping scheme may not lead to plausible reshaping results that reflect the expected deformation of loose-fitting apparel. One approach to circumvent this problem is to create a deformable model that also includes clothes [Stoll et al. 2010], and then use the same for tracking and reshaping. Also, shape edits often lead to corresponding changes in skeletal dimensions. When editing a video, this might make motion retargeting necessary in order to preserve a natural motion (e.g. to prevent foot skating).

Finally, our approach is currently not fully automatic. For segmentation of monocular video we rely heavily on commercial tools that may require manual intervention. Furthermore, we require further user interaction in the form of manual feature point tracks to make ill-posed monocular tracking feasible. In the future, we would like to investigate automatic articulate human detection and pose estimation techniques, along with physically-based constraints to reduce the required interaction.

### 7.2.2 Data-driven 3D Modeling

Not all the shapes generated by our system in Chapter 5 are useful or visually pleasant. Currently, we only provide a very high-level user-interface with very easy-to-use tools; however, our system currently has no tools to repair a shape, e.g., when an undesired placement of a part occurs. One option would be to provide our system as a plug-in for a conventional modeling package. An incorrectly placed part can then be corrected easily with the editing tools provided by the conventional modeling package.

Most of the time, unpleasant or strange-looking recombinations occur if the structure of the source and the target shape are very different or objects of different classes are combined, e. g., a morph between two boats does typically looks more convincing than a morph between a boat and a plane. We would like to build a contact tree from the neighborhood graph, and we believe a shape matching on the tree will generate more plausible results.

The quality of the blended results depends on the way segments are grouped into shapes. If they are coupled loosely, with few contacts (e. g., limbs of robots, or features attached to the body of a vehicle), the approach works best. For too-complex contact constraints, such as segments inside a car, the recombination might fail. Our approach does not allow the deformation of individual parts. Consequently, shapes that have puzzle-like structures (i.e., where each part has to fit into another) typically also produce visible artefacts. In future work, we would also like to perform functional analysis, and we hope this will improve the quality of the intermediate results produced. We would like to extend our part-based decomposition and recombination approach to organic shapes, images, animations, or audio.

### 7.2.3 Data-driven Material Assignment

Our current method is based on the popular Phong reflection model. In future work, we would like to generalize to general BRDFs using a metric such as the one of [Ngan, Durand and Matusik 2006]. We also want to evaluate the results on perceived quality of material assignment when using different reflection models. Currently, our approach assumes the query objects to be segmented into parts. We base this assumption on the fact that while modeling, designers almost always create an object using multiple parts. However, this assumption is not valid for organic objects, e. g., animals or human bodies. We would like to explore the possibility of using automatic segmentation algorithms such as the one proposed by [Kalogerakis, Hertzmann and Singh 2010] and then use the resulting segmented model within our framework. Finally, we would also like to extend our material assignment approach to spatially-varying materials (textures), which would require us to solve the problem of cross-parameterization for query and database shapes.

This thesis argued as to why creating quality content remains difficult, and how statistical models learned from existing visual content databases can enable the next generation content creation tools. We show how data-driven models can lead to more intuitive interfaces which are easier to learn and use even for novice users. At the same time, such tools also enable the user to have precise and flexible control for creating high-quality content.

To bridge the gap between the demand for and supply of such content, combining user interaction with statistical knowledge into new computer graphics algorithms, as this thesis has done, is a promising approach to efficiently generate visual content.

# Bibliography (Own work)


JAIN, A., THORMÄHLEN, T., RITSCHEL, T. AND SEIDEL, H.-P. (2011): Exploring Shape Variations by 3D-Model Decomposition and Part-based Recombination. In *Computer Graphics Forum (Eurographics '11)*

JAIN, A., THORMÄHLEN, T., RITSCHEL, T. AND SEIDEL, H.-P. (2012): Material Memex: Automatic Material Suggestions for 3D Objects. In *ACM Trans. on Graphics (SIGGRAPH Asia '12)*

JAIN, A., THORMÄHLEN, T., SEIDEL, H.-P. AND THEOBALT, C. (2010): MovieReshape: Tracking and Reshaping of Humans in Videos. In *ACM Trans. on Graphics (SIGGRAPH Asia '12)*

PISHCHULIN, L., JAIN, A., ANDRILUKA, M., THORMÄHLEN, T. AND SCHIELE, B. (2012): Learning People Detection Models from Few Training Samples. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR '12)*

PISHCHULIN, L., JAIN, A., WOJEK, C., ANDRILUKA, M., THORMÄHLEN, T. AND SCHIELE, B. (2011a): Articulated People Detection and Pose Estimation: Reshaping the Future. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR '11)*

PISHCHULIN, L., JAIN, A., WOJEK, C., ANDRILUKA, M., THORMÄHLEN, T. AND SCHIELE, B. (2011b): In Good Shape: Robust People Detection based on Appearance and Shape. In *Proc. British Machine Vision Conference (BMVC '11)*

# Bibliography


AGARWAL, A. AND TRIGGS, B. (2006): Recovering 3D Human Pose from Monocular Images. In *IEEE Pattern Analysis and Machine Intelligence (PAMI '06)* 17, 18

AGUIAR, E. DE, STOLL, C., THEOBALT, C., AHMED, N., SEIDEL, H.-P. AND THRUN, S. (2008): Performance Capture from Sparse Multi-View Video. In *ACM Trans. on Graphics (SIGGRAPH '08)* 17

ALEXA, M., COHEN-OR, D. AND LEVIN, D. (2000): As-Rigid-as-Possible Shape Interpolation. In *ACM Trans. on Graphics (SIGGRAPH '00)* 14

ALLEN, B., CURLESS, B. AND POPOVIĆ, Z. (2003): The Space of Human Body Shapes: Reconstruction and Parameterization from Range Scans. In *ACM Trans. on Graphics (SIGGRAPH '03)* 6, 14, 15, 28

ALLEN, B., CURLESS, B., POPOVIĆ, Z. AND HERTZMANN, A. (2006): Learning a Correlated Model of Identity and Pose-Dependent Body Shape Variation for Real-Time Synthesis. In *Proc. Symposium on Computer Animation (SCA '06)* 14

ANGUELOV, D., SRINIVASAN, P., KOLLER, D., THRUN, S., RODGERS, J. AND DAVIS, J. (2005): SCAPE: Shape Completion and Animation of People. In *ACM Trans. on Graphics (SIGGRAPH '05)* 14, 24

BALAN, A. O., SIGAL, L., BLACK, M. J., DAVIS, J. E. AND HAUSSECKER, H. W. (2007): Detailed Human Shape and Pose from Images. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR '07)* 17

BARNES, C., SHECHTMAN, E., FINKELSTEIN, A. AND GOLDMAN, D. B. (2009): PatchMatch: A Randomized Correspondence Algorithm for Structural Image Editing. In *ACM Trans. on Graphics (SIGGRAPH '09)* 39

BARRETT, W. A. AND CHENEY, A. S. (2002): Object-Based Image Editing. In *ACM Trans. on Graphics (SIGGRAPH '02)* 16

BEIER, T. AND NEELY, S. (1992): Feature-Based Image Metamorphosis. In *Proc. SIGGRAPH Comput. Graph. (SIGGRAPH Comput. Graph. '92)* 14

BENNETT, E. P. AND MCMILLAN, L. (2003): Proscenium: a Framework for Spatio-Temporal Video Editing. In *Proc. ACM Multimedia (ACM MM '03)* 16

BERGEN, G. VAN DEN (1998): Efficient Collision Detection of Complex Deformable Models Using AABB Trees. *J. Graph. Tools* 12, 45

BERNER, A., BOKELOH, M., WAND, M., SCHILLING, A. AND SEIDEL, H.-P. (2008): A Graph-Based Approach to Symmetry Detection. In *Proc. Volume and Point-based Graphics* 46

BLANZ, V. AND VETTER, T. (1999): A Morphable Model for the Synthesis of 3D Faces. In *Proc. of the 26th annual conference on Computer graphics and interactive techniques (SIGGRAPH '99)* 6, 14, 15

BOKELOH, M., WAND, M. AND SEIDEL, H. (2010): A Connection Between Partial Symmetry and Inverse Procedural Modeling. In *ACM Trans. on Graphics (SIGGRAPH '10)* 15

BOURDEV, L. AND MALIK, J. (2009): Poselets: Body Part Detectors Trained Using 3D Human Pose Annotations. In *Proc. International Conference on Computer Vision (ICCV'09)* 37

BUSH, V. (1945): As We May Think. *Atlantic Monthly*, 176, 101–108 62

CHAJDAS, M. G., LEFEBVRE, S. AND STAMMINGER, M. (2010): Assisted Texture Assignment. In *Proc. Symposium on Interactive 3D Graphics and Games (I3D '10)* 19

CHAUDHURI, S., KALOGERAKIS, E., GUIBAS, L. AND KOLTUN, V. (2011): Probabilistic Reasoning for Assembly-Based 3D Modeling. In *ACM Trans. on Graphics (SIGGRAPH '11)* 18, 20

CHAUDHURI, S. AND KOLTUN, V. (2010): Data-driven Suggestions for Creativity Support in 3D Modeling. In *ACM Trans. on Graphics (SIGGRAPH Asia '10)* 15

CHEN, D.-Y., TIAN, X.-P., SHEN, Y.-T. AND OUHYOUNG, M. (2003): On Visual Similarity Based 3D Model Retrieval. In *Computer Graphics Forum (Eurographics '03)* 66

COHEN-OR, D., SOLOMOVIC, A. AND LEVIN, D. (1998): Three-Dimensional Distance Field Metamorphosis. In *ACM Trans. on Graphics (SIGGRAPH '98)* 14

DAVIS, J., AGRAWALA, M., CHUANG, E., POPOVIĆ, Z. AND SALESIN, D. (2003): A Sketching Interface for Articulated Figure Animation. In *Proc. Symposium on Computer Animation (SCA '03)* 17

EBERT, D. (2003): Texturing & Modeling: A Procedural Approach. 15

EFROS, A. AND LEUNG, T. (2002): Texture Synthesis by Non-Parametric Sampling. In *Proc. International Conference on Computer Vision (ICCV '02)* 15

ENZWEILER, M. AND GAVRILA, D. M. (2008): A mixed Generative-Discriminative Framework for Pedestrian Classification. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR '08)* 18

FELZENSZWALB, P. F., GIRSHICK, R. B., MCALLESTER, D. AND RAMANAN, D. (2010): Object Detection with Discriminatively Trained Part-Based Models. In *IEEE Pattern Analysis and Machine Intelligence (PAMI '10)* 2

FISHER, M. AND HANRAHAN, P. (2010): Context-Based Search for 3D Models. In *ACM Trans. on Graphics (SIGGRAPH Asia '10)* 20

FISHER, M., SAVVA, M. AND HANRAHAN, P. (2011): Characterizing Structural Relationships in Scenes Using Graph Kernels. In *ACM Trans. on Graphics (SIGGRAPH '11)* 20

FLORIDA, R. (2003): The Rise of the Creative Class: And How It's Transforming Work, Leisure, Community and Everyday Life. 1

FU, H., COHEN-OR, D., DROR, G. AND SHEFFER, A. (2008): Upright Orientation of Man-Made Objects. In *ACM Trans. on Graphics (SIGGRAPH '08)* 19

FUNKHOUSER, T., KAZHDAN, M., SHILANE, P., MIN, P., KIEFER, W., TAL, A., RUSINKIEWICZ, S. AND DOBKIN, D. (2004): Modeling by Example. In *ACM Trans. on Graphics (SIGGRAPH '04)* 15, 18

GAL, R., SORKINE, O., MITRA, N. AND COHEN-OR, D. (2009): iWIRES: An analyze-and-edit approach to shape manipulation. In *ACM Trans. on Graphics (SIGGRAPH '09)* 19

GALL, J., STOLL, C., AGUIAR, E. DE, THEOBALT, C., ROSENHAHN, B. AND SEIDEL, H.-P. (2009): Motion Capture Using Simultaneous Skeleton Tracking and Surface Estimation. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR '09)* 17, 26, 27

GOLOVINSKIY, A. AND FUNKHOUSER, T. (2009): Consistent Segmentation of 3D Models. In *Proc. Shape Modeling International (SMI '09)* 15, 19

GUAN, P., WEISS, A., BĂLAN, A. O. AND BLACK, M. J. (2009): Estimating Human Shape and Pose from a Single Image. In *Proc. International Conference on Computer Vision (ICCV '09)* 17

HASLER, N., ACKERMANN, H., ROSENHAHN, B., THORMÄHLEN, T. AND SEIDEL, H.-P. (2010): Multilinear Pose and Body Shape Estimation of

Dressed Subjects from Image Sets. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR '10)* 17

HASLER, N., STOLL, C., SUNKEL, M., ROSENHAHN, B. AND SEIDEL, H.-P. (2009): A Statistical Model of Human Pose and Body Shape. In *Computer Graphics Forum (Eurographics '09)* 14, 36, 37

HORNUNG, A., DEKKERS, E. AND KOBBELT, L. (2007): Character Animation from 2D Pictures and 3D Motion Data. In *ACM Trans. on Graphics (SIGGRAPH '07)* 16, 17, 81

JACOBSON, A., BARAN, I., POPOVIĆ, J. AND SORKINE, O. (2011): Bounded Biharmonic Weights for Real-Time Deformation. In *ACM Trans. on Graphics (SIGGRAPH'11)* 37, 39

JOHNSON, S. AND EVERINGHAM, M. (2011): Learning Effective Human Pose Estimation from Inaccurate Annotation. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR '11)* 18

KAICK, O. VAN, ZHANG, H., HAMARNEH, G. AND COHEN-OR, D. (2011): A Survey on Shape Correspondence. In *Computer Graphics Forum (Eurographics '11)* 9

KALOGERAKIS, E., HERTZMANN, A. AND SINGH, K. (2010): Learning 3D Mesh Segmentation and Labeling. In *ACM Trans. on Graphics (SIGGRAPH '10)* 15, 19, 20, 82

KAUTZ, J., BOULOS, S. AND DURAND, F. (2007): Interactive Editing and Modeling of Bidirectional Texture Functions. In *ACM Trans. on Graphics (SIGGRAPH '07)* 19

KAZHDAN, M., FUNKHOUSER, T. AND RUSINKIEWICZ, S. (2003): Rotation Invariant Spherical Harmonic Representation of 3D Shape Descriptors. In *Proc. Symposium on Geometry Processing (SGP '03)* 15

KERR, W. B. AND PELLACINI, F. (2010): Toward Evaluating Material Design Interface Paradigms for Novice Users. In *ACM Trans. on Graphics (SIGGRAPH '10)* 19, 73

KILIAN, M., MITRA, N. J. AND POTTMANN, H. (2007): Geometric Modeling in Shape Space. In *ACM Trans. on Graphics (SIGGRAPH '07)* 14

KIM, M., WU, G., YAP, P.-T. AND SHEN, D. (2012): A General Fast Registration Framework by Learning Deformation-Appearance Correlation. In *IEEE Image Processing (IP '12)* 14

KOKKINOS, I. AND YUILLE, A. L. (2007): Unsupervised Learning of Object Deformation Models. In *Proc. International Conference on Computer Vision (ICCV '07)* 14

KRAEVOY, V., JULIUS, D. AND SHEFFER, A. (2007): Model Composition from Interchangeable Components. In *Proc. Pacific Graphics (PG '07)* 18, 56, 59

KRÄHENBÜHL, P., LANG, M., HORNUNG, A. AND GROSS, M. (2009): A system for Retargeting of Streaming Video. In *ACM Trans. on Graphics (SIGGRAPH Asia '09)* 16

KSCHISCHANG, F. R., FREY, B. J. AND LOELIGER, H.-A. (1998): Factor Graphs and the Sum-Product Algorithm. 71

LAGA, H., TAKAHASHI, H. AND NAKAJIMA, M. (2006): Spherical Wavelet Descriptors for Content-based 3D Model Retrieval. In *Proc. Shape Modeling International (SMI '06)* 15

LEE, A. W. F., DOBKIN, D., SWELDENS, W. AND SCHRÖDER, P. (1999): Multiresolution Mesh Morphing. In *ACM Trans. on Graphics (SIGGRAPH '99)* 14

LEE, Y. J., ZITNICK, C. L. AND COHEN, M. F. (2011): ShadowDraw: Real-time User Guidance for Freehand Drawing. In *ACM Trans. on Graphics (SIGGRAPH '11)* 20

LEFEBVRE, S., HORNUS, S. AND LASRAM, A. (2010): By-Example Synthesis of Architectural Textures. In *ACM Trans. on Graphics (SIGGRAPH '10)* 19

LEORDEANU, M. (2009): Spectral Graph Matching, Learning, and Inference for Computer Vision. Ph. D thesis, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 59

LEYVAND, T., COHEN-OR, D., DROR, G. AND LISCHINSKI, D. (2008): Data-driven Enhancement of Facial Attractiveness. In *ACM Trans. on Graphics (SIGGRAPH '08)* 16, 22

LI, W., AGRAWALA, M., CURLESS, B. AND SALESIN, D. (2008): Automated Generation of Interactive 3D Exploded View Diagrams. In *ACM Trans. on Graphics (SIGGRAPH '08)* 19

LI, Y., SUN, J. AND SHUM, H.-Y. (2005): Video Object Cut and Paste. *ACM Trans. on Graphics (SIGGRAPH '05)* 16, 25

LIU, C., TORRALBA, A., FREEMAN, W. T., DURAND, F. AND ADELSON, E. H. (2005): Motion Magnification. In *ACM Trans. on Graphics (SIGGRAPH '05)* 16

LU, J., GEORGHIADES, A. S., GLASER, A., WU, H., WEI, L.-Y., GUO, B., DORSEY, J. AND RUSHMEIER, H. (2007): Context-Aware Textures. *ACM Trans. on Graphics (SIGGRAPH '07)* 19

MALISIEWICZ, T. AND EFROS, A. A. (2009): Beyond Categories: The Visual Memex Model for Reasoning About Object Relationships. In *Proc. Neural Information Processing Systems (NIPS '09)* 20, 62

MARIN, J., VAZQUEZ, D., GERONIMO, D. AND LOPEZ, A. (2010): Learning Appearance in Virtual Scenarios for Pedestrian Detection. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR '10)* 18

MEHRA, R., ZHOU, Q., LONG, J., SHEFFER, A., GOOCH, A. AND MITRA, N. (2009): Abstraction of Man-Made Shapes. In *ACM Trans. on Graphics (SIGGRAPH '09)*, 137 19

MERRELL, P. (2007): Example-Based Model Synthesis. In *Proc. Symposium on Interactive 3D Graphics and Games (I3D '07)* 15

MERTENS, T., KAUTZ, J., CHEN, J., BEKAERT, P. AND DURAND, F. (2006): Texture Transfer Using Geometry Correlation. In *Proc. Eurographics Symposium on Rendering (EGSR '06)* 19

MITRA, N. J., GUIBAS, L. AND PAULY, M. (2007): Symmetrization. In *ACM Trans. on Graphics (SIGGRAPH '07)* 14

MITRA, N., GUIBAS, L. AND PAULY, M. (2006): Partial and approximate symmetry detection for 3D geometry. In *ACM Trans. on Graphics (SIGGRAPH '06)* 19

MITRA, N., YANG, Y., YAN, D., LI, W. AND AGRAWALA, M. (2010): Illustrating how Mechanical Assemblies Work. In *ACM Trans. on Graphics (SIGGRAPH '10)* 19

MÜLLER, M., HEIDELBERGER, B., HENNIX, M. AND RATCLIFF, J. (2007): Position Based Dynamics. *J. Vis. Comm. Image Rep.* 55

MÜLLER, M., HEIDELBERGER, B., TESCHNER, M. AND GROSS, M. (2005): Meshless Deformations Based on Shape Matching. *ACM Trans. on Graphics (SIGGRAPH '05)* 29, 31

NEWCOMBE, R. A., DAVISON, A. J., IZADI, S., KOHLI, P., HILLIGES, O., SHOTTON, J., MOLYNEAUX, D., HODGES, S., KIM, D. AND FITZGIBBON, A. (2011): KinectFusion: Real-time Dense Surface Mapping and Tracking. In *Proc. Mixed and Augmented Reality (ISMAR '11)* 80

NGAN, A., DURAND, F. AND MATUSIK, W. (2006): Image-Driven Navigation of Analytical BRDF Models. In *Proc. Eurographics Symposium on Rendering (EGSR '06)* 19, 82

OKADA, R. AND SOATTO, S. (2008): Relevant Feature Selection for Human Pose Estimation and Localization in Cluttered Images. In *Proc. European Conference on Computer Vision (ECCV '08)* 18

OSADA, R., FUNKHOUSER, T., CHAZELLE, B. AND DOBKIN, D. (2002): Shape Distributions. *ACM Trans. on Graphics (SIGGRAPH '02)* 15

OVSJANIKOV, M., BRONSTEIN, A., BRONSTEIN, M. AND GUIBAS, L. (2009): Shape Google: A Computer Vision Approach to Isometry Invariant Shape Retrieval. In *Proc. International Conference on Computer Vision Workshops (ICCV Workshops '09)* 15

OVSJANIKOV, M., LI, W., GUIBAS, L. AND MITRA, N. J. (2011): Exploration of Continuous Variability in Collections of 3D Shapes. In *ACM Trans. on Graphics (SIGGRAPH '11)* 15, 19

PARAMESWARAN, V. AND CHELLAPPA, R. (2004): View Independent Human Body Pose Estimation from a Single Perspective image. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR '04)* 17

PAULY, M., MITRA, N. J., WALLNER, J., POTTMANN, H. AND GUIBAS, L. (2008): Discovering Structural Regularity in 3D Geometry. In *ACM Trans. on Graphics (SIGGRAPH '08)* 19

PELLACINI, F., FERWERDA, J. A. AND GREENBERG, D. P. (2000): Toward a Psychophysically-Based Light Reflection Model for Image Synthesis. In *ACM Trans. on Graphics (SIGGRAPH '00)* 67

PELLACINI, F. AND LAWRENCE, J. (2007): AppWand: Editing Measured Materials Using Appearance-Driven Optimization. In *ACM Trans. on Graphics (SIGGRAPH '07)* 19

PHONG, B. T. (1975): Illumination for Computer Generated Pictures. *Commun. ACM* 19, 64

POPPE, R. (2007): Vision-Based Human Motion Analysis: An Overview. In *Computer Vision and Image Understanding (CVIU '07)* 17

RAMANAN, D. (2006): Learning to Parse Images of Articulated Objects. In *Proc. Neural Information Processing Systems (NIPS '06)* 36

RITSCHEL, T., OKABE, M., THORMÄHLEN, T. AND SEIDEL, H.-P. (2009): Interactive Reflection Editing. In *ACM Trans. on Graphics (SIGGRAPH Asia '09)* 31

ROSALES, R. AND SCLAROFF, S. (2006): Combining Generative and Discriminative Models in a Framework for Articulated Pose Estimation. In *Int. J. Comput. Vision (IJCV '06)* 17

ROTHER, C., KOLMOGOROV, V. AND BLAKE, A. (2004): "GrabCut": interactive foreground extraction using iterated graph cuts. In *ACM Trans. on Graphics (SIGGRAPH '04)* 37

RUBINSTEIN, M., SHAMIR, A. AND AVIDAN, S. (2008): Improved Seam Carving for Video Retargeting. In *ACM Trans. on Graphics (SIGGRAPH '08)* 16

SCHAEFER, S., MCPHAIL, T. AND WARREN, J. (2006): Image Deformation Using Moving Least Squares. In *ACM Trans. on Graphics (SIGGRAPH '06)* 29

SCHOLZ, V. AND MAGNOR, M. (2006): Texture Replacement of Garments in Monocular Video Sequences. In *Proc. Eurographics Symposium on Rendering (EGSR '06)*, 305–312 16, 22

SCHOLZ, V., EL-ABED, S., SEIDEL, H.-P. AND MAGNOR, M. A. (2009): Editing Object Behaviour in Video Sequences. In *Computer Graphics Forum (Eurographics '09)* 16, 22

SEO, H. AND MAGNENAT-THALMANN, N. (2004): An Example-Based Approach to Human Body Manipulation. *Graph. Models* 14

SHAKHNAROVICH, G., VIOLA, P. AND DARRELL, T. (2003): Fast Pose Estimation with Parameter Sensitive Hashing. In *Proc. International Conference on Computer Vision (ICCV '03)* 18

SHARF, A., BLUMENKRANTS, M., SHAMIR, A. AND COHEN-OR, D. (2006): SnapPaste: An Interactive Technique for Easy Mesh Composition. *The Visual Computer* 19

SHOTTON, J., FITZGIBBON, A., COOK, M., SHARP, T., FINOCCHIO, M., MOORE, R., KIPMAN, A. AND BLAKE, A. (2011): Real-Time Human Pose Recognition in Parts from a Single Depth Image. In *Proc. Conference on Computer Vision and Pattern Recognition (CVPR '11)* 18

SIGAL, L., BALAN, A. O. AND BLACK, M. J. (2007): Combined Discriminative and Generative Articulated Pose and Non-Rigid Shape Estimation. In *Proc. Neural Information Processing Systems (NIPS '07)* 17

SNAVELY, N., SEITZ, S. M. AND SZELISKI, R. (2006): Photo Tourism: Exploring Photo Collections in 3D. In *ACM Trans. on Graphics (SIGGRAPH '06)* 80

STOLFI, J. (1991): Oriented Projective Geometry: A Framework for Geometric Computation. 26

STOLL, C., GALL, J., AGUIAR, E. DE, THRUN, S. AND THEOBALT, C. (2010): Video-Based Reconstruction of Animatable Human Characters. In *ACM Trans. on Graphcics (SIGGRAPH Asia '10)* 81

SUMNER, R. W., ZWICKER, M., GOTSMAN, C. AND POPOVIĆ, J. (2005): Mesh-based Inverse Kinematics. In *ACM Trans. on Graphics (SIGGRAPH '05)* 14

TORRALBA, A. (2003): Contextual Priming for Object Detection. *Int. J. Comput. Vision (IJCV '03)* 62

VLASIC, D., BARAN, I., MATUSIK, W. AND POPOVIĆ, J. (2008): Articulated Mesh Animation from Multi-View Silhouettes. In *ACM Trans. on Graphics (SIGGRAPH '08)* 17

VLASIC, D., BRAND, M., PFISTER, H. AND POPOVIĆ, J. (2005): Face Transfer with Multilinear Models. In *ACM Trans. on Graphics (SIGGRAPH '05)* 16, 22

WANG, B., YU, Y., WONG, T.-T., CHEN, C. AND XU, Y.-Q. (2010): Data-Driven Image Color Theme Enhancement. In *ACM Trans. on Graphics (SIGGRAPH Asia '10)* 20

WANG, H., XU, N., RASKAR, R. AND AHUJA, N. (2007): Videoshop: A New Framework for Spatio-Temporal Video Editing in Gradient Domain. *Graph. Models* 16

WANG, J., BHAT, P., COLBURN, R. A., AGRAWALA, M. AND COHEN, M. F. (2005): Interactive Video Cutout. In *ACM Trans. on Graphics (SIGGRAPH '05)* 16, 25

WANG, J., DRUCKER, S. M., AGRAWALA, M. AND COHEN, M. F. (2006): The cartoon Animation Filter. In *ACM Trans. on Graphics (SIGGRAPH '06)* 16

WANG, Y., XU, K., LI, J., ZHANG, H., SHAMIR, A., LIU, L., CHENG, Z. AND XIONG, Y. (2011): Symmetry Hierarchy of Man-Made Objects. In *Computer Graphics Forum (Eurographics '11)* 19

WEI, X. AND CHAI, J. (2010): VideoMocap: Modeling Physically Realistic Human Motion From Monocular Video Sequences. In *ACM Trans. on Graphics (SIGGRAPH '10)* 17

WOLBERG, G. (1998): Image morphing: A survey. *The Visual Computer* 14

XU, K., LI, H., ZHANG, H., COHEN-OR, D., XIONG, Y. AND CHENG, Z.-Q. (2010): Style-Content Separation by Anisotropic Part Scales. In *ACM Trans. on Graphics (SIGGRAPH Asia '10)* 9

XU, W., WANG, J., YIN, K., ZHOU, K., VAN DE PANNE, M., CHEN, F. AND GUO, B. (2009): Joint-Aware Manipulation of Deformable Models. In *ACM Trans. on Graphics (SIGGRAPH '09)* 19

YU, L.-F., YEUNG, S. K., TANG, C.-K., TERZOPOULOS, D., CHAN, T. F. AND OSHER, S. (2011): Make it Home: Automatic Optimization of Furniture Arrangement. In *ACM Trans. on Graphics (SIGGRAPH '11)* 20

ZHENG, Y., FU, H., COHEN-OR, D., AU, O. K.-C. AND TAI, C.-L. (2011): Component-Wise Controllers for Structure-Preserving Shape Manipulation. In *Computer Graphics Forum (Eurographics '11)* 19

ZHOU, S., FU, H., LIU, L., COHEN-OR, D. AND HAN, X. (2010): Parametric Reshaping of Human Bodies in Images. In *ACM Trans. on Graphics (SIGGRAPH '10)* 16, 17, 18, 22