# Photorealistic Augmented Reality

Simon Gibson[1], Alan Chalmers[2]

[1] Advanced Interfaces Group, University of Manchester, UK,
[2] Department of Computer Science, University of Bristol, UK.

**Abstract**

*Augmenting real-world images with synthetic objects is becoming of increasing importance in both research and commercial applications, and encompasses aspects of fields such as mobile camera and display technology, computer graphics, image processing, computer vision and human perception. This tutorial presents an in-depth study into the techniques required to produce high fidelity augmented images at interactive rates, and will consider how the realism of the resulting images can be assessed and their fidelity quantified.*
*The first half of the tutorial covers the methods we use to generate augmented images. We will show how commonly available digital cameras can be used to record scene data, and how computer graphics hardware can be used to generate visually realistic augmented images at interactive rates. Specific topics covered will include geometric and radiometric camera calibration, image-based reconstruction of scene geometry and illumination, hardware accelerated rendering of synthetic objects and shadows, and image compositing. The second half of the tutorial discusses in more detail what we are trying to achieve when generating augmented images, and how success can be measured and quantified. Methods for displaying augmented images will be discussed, and techniques for conducting psychophysical experiments to evaluating the visual quality of images will also be covered.*
*Examples of augmented images and video sequences from a real-world interactive interior design application will be shown, and used to illustrate the different ideas and techniques introduced throughout the tutorial.*

Categories and Subject Descriptors (according to ACM CCS): H.5.1 [Information Interfaces and Presentation]: Multimedia Information SystemsArtificial, augmented, and virtual realities I.3.3 [Computer Graphics]: Picture/Image GenerationBitmap and framebuffer operations I.3.7 [Computer Graphics]: Three-Dimensional Graphics and RealismColor, shading, shadowing and texture.

## 1. Introduction

The ability to merge synthetically generated objects into images of a real scene is becoming central to many applications of computer graphics, and in particular, mixed or augmented reality. In many situations, this merging must be done at rates of many frames-per-second if an illusion of interactivity is to be maintained. Also, visually realistic combinations of objects and background images are required if the ultimate goal of augmentation is to present images to the user that are indistinguishable from reality. To achieve these goals the synthetic objects must be registered both *geometrically* and *photometrically* with the camera. Geometric registration is required to orient the synthetic object to the same perspective, and composite with the background image. Photometric registration is required to ensure that synthetic objects are illuminated using the same lighting conditions as in the real scene. Finally, the augmentation process requires determining how the synthetic objects affect the illumination *already present* in the scene. Typically, these changes in illumination take the form of reflections of the synthetic object in background surfaces, and occlusions of light transport paths that manifest themselves as shadows cast onto the real objects.

Traditionally, the competing requirements of real-time rendering and visual realism have meant that generating photorealistic augmented images at interactive rates has been a distant goal. Recently however, techniques have been developed that allow synthetic objects to be illuminated by complex lighting environments in real-time (see for example [44, 60, 69, 27, 51]). In this tutorial, we will show how techniques like these can be used to generate visually realis-

tic augmented images at interactive rates using commonly available graphics hardware. We will also describe the techniques we employ to capture geometric and photometric data from the scene. We will also addresses three important issues concerned with the perception of rendered and augmented images:

- How realistic are the synthesised images?
- How does the display device affect the perceived realism of the images?
- How can we judge the quality of the images the user will perceive?

We will look at how the perceptual quality of images can be measured using both perceptual and numerical techniques. Tone mapping operators are described for overcoming some of the limitations of current display technology. Finally, important issues are raised that must be considered when preparing any psychophysical experiments to assess image quality.

### 1.1. The ARIS Project

Much of the work presented in these notes comes from the ARIS project[1]. The goal of the ARIS project is to provide new technologies for the seamless integration of virtual objects into augmented environments, and to develop new visualisation and interaction paradigms for novel collaborative AR applications. Two different application scenarios are being developed:

- An interactive desktop system, where the end-user can easily integrate 3D product models (e.g. furniture) into a set of images of his real environment, taking consistent illumination of real and virtual objects into account.
- A mobile AR-unit, where 3D product models can be directly visualised on a real site and be discussed with remote participants, including new collaborative and shared augmented technologies.

Both approaches are being tested and validated in end-user trials, addressing the new application area of e-commerce. In addition to existing e-commerce solutions, where mainly product catalogues can be listed, the ARIS project is aiming to enable the presentation of products in the context of their future environments (e.g. new furniture for a living room, new light sources for an office, etc.).

The interactive ARIS-system allows the user to reconstruct geometric and illumination properties from of a set of images in a semi-automatic way. The user can place 3D product models in the reconstructed image space and see the direct and indirect lighting effects in his real environment caused by the virtual modifications at interactive update rates.

Figure 1 shows some example images taken from an environment typical of those found within the ARIS project. This is intended to illustrate some of the problems that must be overcome before visually realistic augmented images can be produced. In the top-left is an image of an empty scene that will be augmented with furniture items. When a real chair is introduced into the room (top-right), it is illuminated by the same light as the rest of the scene, but also interrupts the passage of light from the window to the floor, resulting in a series of shadows cast onto the scene. The shadows cast by the object may fall onto any "real" surface in the scene, and the scene itself must also cast shadows onto the objects, affecting both their shading and the shadows that they cast (middle row).

### 1.2. Overview

Figure 2 shows an example of the rendering process that will be described in more detail in the first half of this tutorial. We will discuss in detail the following three stages of image generation:

- The geometric registration of synthetic objects into a background image. We will discuss how a single photograph of a scene can be used to both calibrate the camera position and generate an approximate 3D representation of the scene. This is then used to perform a depth-composite of the synthetic objects with the background photograph. Real-time camera registration, although not discussed in detail here, is mentioned in Section 11.
- The reconstruction of illumination data using a light-probe[17] and high dynamic-range imaging[18]. We will describe how high dynamic range images of the light-probe may be captured, and projected onto the reconstructed 3D representation of the scene in order to generate a *radiance mesh*.
- The use of the radiance mesh to shade synthetic objects and generate their shadows. An irradiance volume and dynamic environment-mapping is employed, along with hardware-accelerated shadow mapping to approximate soft shadows cast by the real lighting environment.

The first half of this tutorial describes these steps in more detail. Section 2 gives a summary of related work in this field, and Section 3 describes the techniques we use to calibrate images and construct a 3D geometric model of the scene. Section 4 then presents the approach we use to capture real-world illumination, including radiometric camera calibration. Section 5 describes how an irradiance volume can be used to illuminate synthetic objects by the reconstructed lighting information, and how environment maps may be generated dynamically from the radiance mesh in order to approximate specular reflections. Section 6 the presents an algorithm that uses commodity graphics hardware to approximate soft shadows cast by synthetic objects, and shows how the shadows can be composited with the background image. Results for a variety of lighting environments and shadow types are given in Section 7, along with visual comparisons between our algorithm, ray-traced images and photographs.

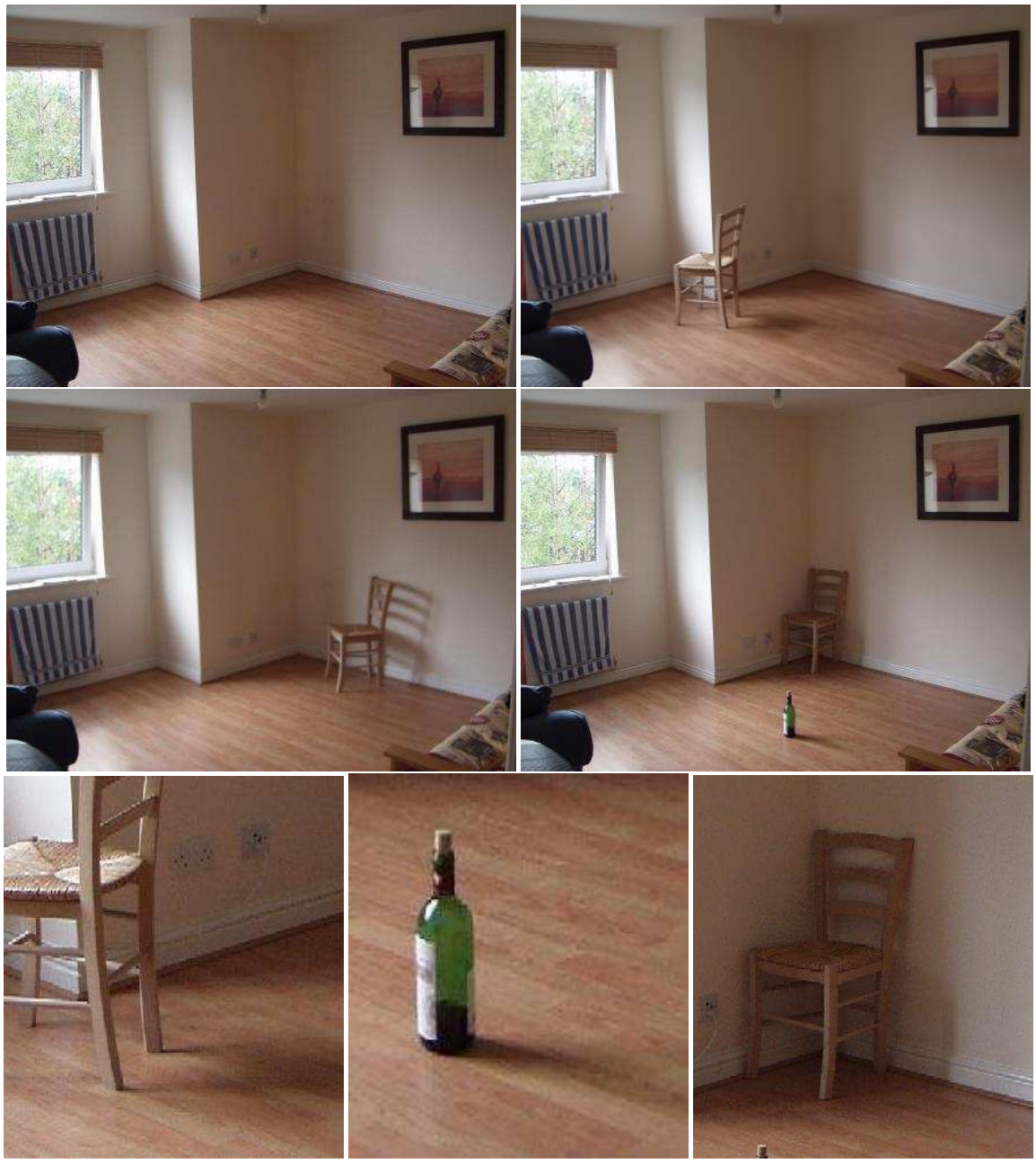


**Figure 1:** *Some real photographs of a scene, illustrating some of the effects that must be captured in order to achieve a visually realistic scene augmentation. The empty scene is shown at the top-left, with a window illuminating the scene. Additional objects must be illuminated by the same light (top-right), and both cast shadows onto the scene (middle-left) and have shadows cast onto them by the scene(middle-right). Closeup views are shown on the bottom row, illustrating both diffuse and specular shading and shadows.*

We also illustrate the graceful trade-off between image quality and rendering time we can achieve.

The second half of these notes cover issues relating to the problem of assessing image quality using psychophysics and the human visual system. Section 8 gives a brief in-

**Figure 2:** *Overview of the rendering process. From the calibrated camera position corresponding to the background image (top), the reconstructed scene geometry is drawn into the depth buffer and synthetic objects are rendered into the color and depth buffers to resolve occlusions (middle). Finally, multiple shadow passes are performed, blending shadows into the composited image (bottom). For this example, the image on the right was generated at over 9 frames per second.*

troduction to the area of visual perception, as it relates to computer graphics, and includes a review of suitable image quality metric for assessing image fidelity. Following that, Section 9 describes in more detail the problem of displaying real images (or simulations of real images) on computer displays. Section 10 then describes some of the issues that must be addressed when designing psychophysical experiments to assess image fidelity. Finally, we draw conclusions and summarise the work in Section 11.

## 2. Related Work

### 2.1. Scene Modelling

Our augmented reality system requires that we have access to a 3D model of the scene. Building 3D models that resemble real environments has always been a difficult problem. Traditional methods of constructing models have involved a skilled user and 3D CAD (Computer Aided Design) software. Accurately modelling a real environment in such a way can only be done if the user has obtained maps or blueprints of the scene, or has access to the scene in order to take precise physical measurements. Either way, the process is slow and laborious for anything but the simplest of scenes.

In the field of computer vision, automatic techniques have recently been developed that allow three-dimensional information to be constructed directly from photographs of the scene[4, 24, 38, 55]. Typically, these algorithms analyse multiple images of an environment in order to infer the position and attributes of each camera, as well as the three-dimensional location of a dense set of points corresponding to important features in the images. In order to build more useful polygonal models, these points must be triangulated and subsequently segmented into separate objects. Similar triangulation and segmentation algorithms are required when expensive laser-range scanners are used to sample the scene geometry[82, 3].

Automatic reconstruction techniques are, however, typically not yet robust enough to build useful models, which must have a simple enough form to allow them to be rendered in real-time, and yet must contain enough well defined structure so that occlusions can be correctly resolved within the Augmented environment. Additionally, automatic algorithms are able only to reconstruct geometry that is seen explicitly in the images, and require more than one source image to work correctly. This causes problems when the user wishes to augment a single photograph of a scene.

In order to overcome problems such as these, semi-automatic approaches have been proposed that employ user-assistance to help with the calculation of the position of those objects and the camera parameters[5, 19, 8, 12, 56, 14, 37, 16]. The benefit of using semi-automatic, rather than fully automatic algorithms, is that we can employ user knowledge when modelling the environment: the walls of a room may be identified by the user and modelled as single large polygons, thereby overcoming problems caused by object occlusion. An object hierarchy may also be easily maintained during the construction of the scene and environments may be constructed in an incremental fashion, with large features specified at the start of the construction process and extra details added as necessary depending upon the envisaged use of the model.

### 2.2. Augmented Image Synthesis

There has been an enormous amount of research devoted to image synthesis and shadow generation. The literature is too large to review in these notes, but useful surveys can be found in[15, 66, 32, 80, 36]. Here, we will focus on previous work that is related to the problem of shading objects in realtime, generating realistic shadows at interactive rates, and composite synthetic objects with real images.

In order to illuminate synthetic objects with real light,

researchers have employed various techniques, ranging from image-based lighting and monte-carlo ray-tracing[17, 64] to texture-based approaches using computer graphics hardware[44]. Recent approaches also use low order spherical harmonics to store precomputed radianceof radiance transport, allowing objects to be shaded and self-shadowed in real-time within arbitrary low-order lighting environments[69, 68]. Another related approach that has been proposed is to use an irradiance volume to shade synthetic objects[34]. This will be discussed in more detail in Section 5.

Basic shadow-mapping techniques[79] have been extended to generate soft shadows by approximating the penumbral regions using several hard-edged shadow[10]. By rendering each shadow from a slightly different position on the light source, and then combining the maps together, realistic representations of soft shadows can be generated. Alternative approaches that attempt to reduce the cost of soft shadow generation include convolution[70], "soft objects"[52] or search techniques[9] to approximate the penumbral region.

Radiosity[15, 66] has previously been used to generate soft shadows, but at a large computational cost. More recently, extensions to these techniques have been made to allow updates to localised regions of the solution, allowing for object movement (see, for example[22, 33, 73]). Following pioneering work by Fournier *et al.*[25], Drettakis *et al.*[21] and Loscos *et al.*[45] used an interactive cluster-based radiosity system to generate the shadows cast by a synthetic object in a real-environment, and composited those shadows into a background photograph at rates of $1-2$ frames per second. Keller has also introduced the "Instant Radiosity" algorithm[43] that uses shadow-mapping hardware to accelerate the generation of globally illuminated environments.

The difficulty in applying hardware-based shadow-mapping to photorealistic Augmented Reality lies in the fact that real-world lighting environments contain a wide variety of different types of light sources, ranging from small focused spot-lights to broad area lights or even diffuse skylight. As the number or area of light sources increases, it becomes harder to apply shadow-mapping and generate believable synthetic shadows. This is especially so if important secondary sources of illumination are required to cast shadows.

To deal with the problem of rendering with a wide variety of real-world light sources, Debevec proposed the use of *image-based lighting* techniques to allow real-world lighting environments to be captured and used to illuminate synthetic objects[17]. High dynamic-range images[18] of a light probe were used in conjunction with a ray-tracing algorithm to render shadows cast by synthetic objects. Differential rendering techniques (discussed in more detail in Section 6.3) were used to produce photorealistic augmented images containing caustics and shadows. A similar algorithm was proposed by Sato *et al.*[64], with the light probe replaced by a camera with a hemispherical lens. Unfortunately, due to the compute intensive nature of the ray-tracing algorithms used in these approaches, interacting with the synthetic objects at rates required in Augmented Reality applications is not yet possible.

To achieve interactive update rates whilst rendering with real-world illumination, Gibson and Murta proposed using computer graphics hardware to render the shadows cast by synthetic objects[30]. Shadows were approximated using multiple hard-edged shadow-maps, and blended into the background image using accumulation-buffer hardware. Although capable of generating images at rates of several frames-per-second, their approach assumed that all light sources in the scene were distant from the synthetic objects. Shadows cast by the objects were also only valid when falling onto a horizontal surface lying immediately below the object, limiting the applicability of the algorithm.

Unlike the techniques described in[30], the shadow generation algorithm used in this work is not constrained by the assumption of distant light sources, allowing for more general lighting environments to be used. Shadows cast by the synthetic objects are also accurate for all orientations and positions of receiver surface. Finally, our algorithm is capable of trading accuracy against rendering time, enabling synthetic objects and subjectively realistic shadows to be merged into background images in real time.

## 3. Scene Modelling

Geometric and Photometric scene reconstruction is achieved using a combination of image-based modeling and high dynamic-range (HDR) imaging techniques. The procedure starts with capturing a single low dynamic-range (LDR) image of the environment, which will be use as the image we augment with synthetic objects. In order to accurately register objects into this image, the position and intrinsic parameters of the camera must be estimated.

Camera calibration is achieved with the aid of user-defined *vanishing points*[13]. An example is shown in Figure 3, where the user has marked edges parallel to the X and Y axes of the required coordinate system (shown with red and green lines respectively). Assuming each pair of edges is not parallel, they can be intersected in image space and used to estimate the camera focal length, position, and orientation. Ideally, three pairs of vanishing points should be marked, one pair for the X, Y and Z axes. However, if it is assumed that the camera's principal point[39] is located in the centre of the image, then two pairs suffice. More specifically, if $v$ and $v'$ are the image-space coordinates of two orthogonal vanishing points, the camera focal length can be determined as follows:

$$f^2 = -v_x v'_x - v_y v'_y \tag{1}$$

After the camera's focal length has been estimated, the position and orientation of the camera can also be found.

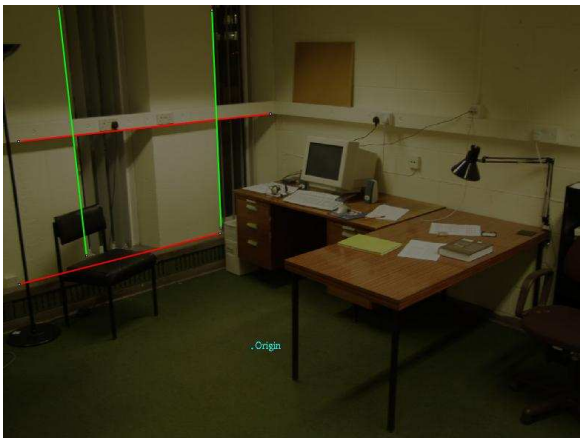



**Figure 3:** *Vanishing points are identified by marking parallel edges in the image. For this scene, the user has marked two green edges that are parallel to the vertical direction, and two red edges that are orthogonal to this direction. An origin point has also been placed in the lower-middle area of the image.*

Camera orientation is found by using the inverse of the camera calibration matrix to transform image-space vanishing points into direction vectors. Provided that the use has marked orthogonal edges in the image, the two vanishing point direction vectors will also be orthogonal in object space. If the third vanishing point has not been marked, it can be estimated from the other two direction vectors. These three orthogonal vectors can then be used to form the columns of the camera rotation matrix. Camera position is set by having the user mark an arbitrary origin point in the image. A direction vector through this point is then found, and the camera position set to lie along this vector. The overall scale of the scene can be fixed, if required, by adjusting the distance of the camera position from the origin.

Once the camera calibration data has been obtained, the process of interactive model reconstruction can begin. We use a simple image-based modelling algorithm presented in[29], which is able to work from single or multiple images, and even video sequences. The user builds the model by interactively specifying the position, orientation and size of objects from a user-extensible library of shapes. Primitive manipulation is achieved "through the lens", by adjusting the projections of each primitive in the image plane. As these primitives are created, a scene-graph is maintained that describes the layout of the scene. The user manipulates these primitives in image space, attempting to match them to objects visible in the photograph. Manipulations of each object in image space are mapped into object space using a set of user-specified constraints and a non-linear optimization algorithm.

Two types of constraints are used to assist the user in primitive manipulation: hierarchical constraints are strictly enforced and affect the position of one primitive with respect to its parent in the scene graph. Image-based constraints, on the other hand, are less strictly enforced and indicate image locations onto which primitive vertices should project. As the user manipulates these constraints, the non-linear optimization algorithm updates the parameters of the primitives so that all hierarchical constraints are satisfied exactly, and all image-based constraints are satisfied as accurately as possible (see[29] for further details).

An example of the reconstruction procedure is shown in Figure 4. In the top-left image, a single box primitive has been created by the user. This box will be manipulated to model the floor, walls and ceiling of the room.

The user interactively adjusts the position of the primitive's vertices so that the edges are aligned with the walls and floor of the room (top-right). The box primitive is constrained to sit on top of the ground-plane by a hierarchical constraint. As each vertex is moved, the optimization algorithm changes the position or orientation of the box in object space, so that its corner vertices project onto the image plane at the positions indicated by the user. As each vertex is placed into its final position, image constraints are created at these locations. When the user selects another vertex and changes its position, the optimization algorithm attempts to satisfy both the projection to the mouse location and the previous image constraint. In this case, this may also involve altering the primitive's size.

As further primitives are created, a scene-graph is incrementally constructed that specifies the relationship between the objects in the scene. The position of one primitive with respect to its parent may be restricted using hierarchical constraints. By default, a primitive is constrained so that the bottom face of its bounding box sits on top of the top face of its parent's bounding box. When modelling from a single image, this allows us to construct the geometric model "from the ground up", thereby ensuring that all objects are represented with a consistent scale (middle row). These constraints are easily changed so that, for instance, an object may be placed on the right-hand side of its parent. As primitives are manipulated, the non-linear optimization algorithm is applied recursively up and down the scene-graph. Because the hierarchical and image constraints attempt to fix a primitive in space, recursion can almost always be limited to at most one level above or below a primitive. The small number of parameters that must be estimated for each movement ensure that the minimization algorithm is typically able to run in real-time as the user adjusts a primitive. The primitive parameters for the last adjustment are then used as the starting guess for the next optimization. The parameter changes required from frame-to-frame are typically small and this helps the optimization algorithm converge quickly to the desired result. Further details of these techniques are given in[29].

A more complete example of the reconstruction, built

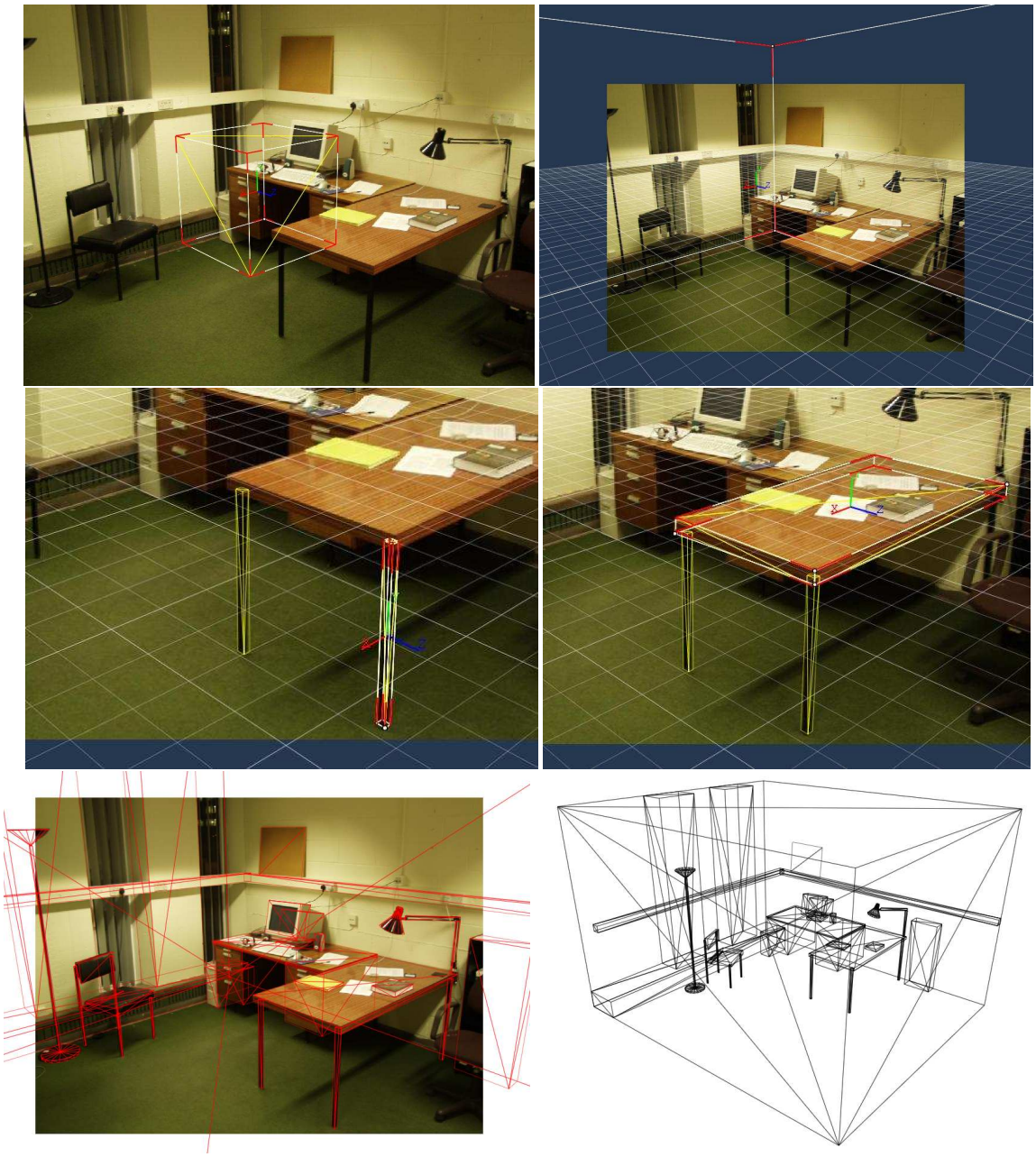


**Figure 4:** *Snapshots taken during the geometry reconstruction phase, showing the interactive reconstruction of an environment from a single photograph. Starting from the ground-up, the 3D geometry is reconstructed using simple parameterized primitives (see text for details). Bottom row shows a view of the entire 3D model. Note that the geometry of the scene has been approximated very roughly in areas not visible in the original photograph.*

from several dozen primitives is shown in the bottow left of Figure 4. Reconstruction time for this simple model was around thirty minutes. Note that we have only modelled the most significant pieces of scene geometry.

## 4. Illumination Capture

Once an approximate representation of the scene geometry has been obtained, it can be used to assist in the reconstruction of illumination properties. This is achieved by captur-

**Figure 5:** *Taking pictures at multiple exposure times. The full dynamic range of the scene is captured in multiple images. For example, detail in very bright parts of the scene is captured by short exposures, and detail in darker parts by longer exposures. Data from all exposures is merged together to form a single High Dynamic-Range image.*

ing high dynamic range (HDR) images of the scene that encode the full dynamic range of light. These images are then processed, and the resulting data used to illuminate the synthetic objects. Most computer graphics software works in a 24 bit RGB space with 8 bits allocated to each of the three primaries. The advantage of this is that no tone mapping is required and the result can be accurately reproduced on a standard CRT. The disadvantage is that colors outside the sRGB gamut cannot be represented (especially very light or dark ones).

### 4.1. Generating High Dynamic Range Imaging

There are two main methods for generating HDR images. The first method is by using physically based renderers which produce high dynamic range images generating basically all visible colors. Another way to generate HDR imaging is by taking photographs of a particular scene at different exposure times[18]. By taking a series of photographs at different exposures, all the luminances in the scene can be captured as shown in Figure 5. After the images have been aligned geometrically to compensate for slight camera movement between each exposure, the camera response function can be recovered using the techniques described in[18, 49]. The response function described how exposure (being a product of exposure time and irradiance at the camera sensor) is related to pixel intensity for each of the red, green and blue channels in the image. An example response function is shown in Figure 6.

Once, the response function is known, information from the multiple exposures may be merged together to form a single high-dynamic range image. Note also that after the response function has been estimated once for a particular camera, it can be used to transform any set of exposures taken with that camera into the high dynamic-range format. In situations like that described below, where the HDR image is going to be used to illuminate synthetic objects, it is important to make sure that the high-end of the dynamic range is captured accurately (e.g. the bright light-sources). Often, it is possible to do this from a single exposure, taken to ensure that the bright light-sources in the scene are not

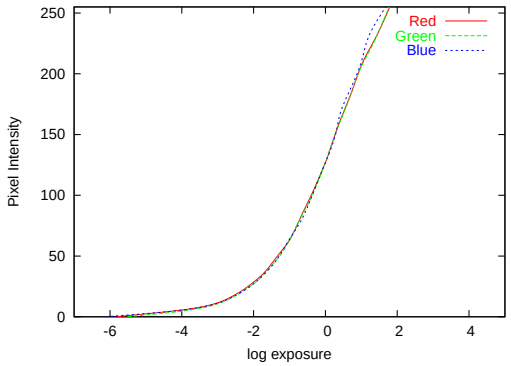



**Figure 6:** *A typical camera response function, showing the relationship between radiance pixel intensity and exposure (the product of irradiance and exposure time).*

clamped at the top of the displayable range. Alternatively, the automatic exposure bracketing found on many cameras can be used to capture different exposure times using a single button press.

### 4.2. Lighting Reconstruction

The overall approach to capturing lighting data is illustrated in Figure 7. A light probe[17] and simple calibration grid are placed in the scene, and a second LDR image is captured from the same camera position. This is the image that is used to construct an approximate model of the environment, using the techniques described in the previous section. Importantly, the calibration grid is also modelled using a square polygon, allowing us to position the light probe relative to the reconstructed scene model. The camera is then moved and a close-up HDR image of the light probe and calibration grid is captured.

Further vanishing-point estimation using the calibration grid in the HDR light probe image allows the position of the probe to be calculated relative to the grid. Because the grid has also been located during geometry reconstruction,

the position of the light probe relative to the scene model can be calculated. A triangular patch mesh is built over the surfaces of the model, and radiance information is projected outwards from the light probe and stored with each patch (Figure 7, bottom).

For our original LDR image, we can now calculate the closest point visible from the camera position through each pixel. Assuming our reconstructed surfaces emit and reflect light diffusely, we can estimate the total irradiance at each point. Using the inverse of the camera response function, we can also map each LDR pixel intensity to a radiance value, and therefore obtain an approximate diffuse reflectivity for each pixel in the LDR image by calculating the ratio of pixel radiance to total irradiance. Each patch in the mesh is assigned an average reflectivity and radiance value, based on its pixel coverage. Finally, reflectivities and radiances at patch vertices are estimated by averaging the values associated with incident patches.

For those surfaces that are not visible in the light probe image, we assign an approximate reflectivity and gather irradiance from the patch mesh, using ray-casting to evaluate visibility[15]. This provides approximate radiance values for the missing surfaces. Again, we assume all surfaces are diffuse, and use an approximate reflectivity of $(0.5, 0.5, 0.5)$. Although more complex inverse illumination algorithms[81, 6] could be used, we have found that these simple approximations are sufficiently accurate for the task in hand.

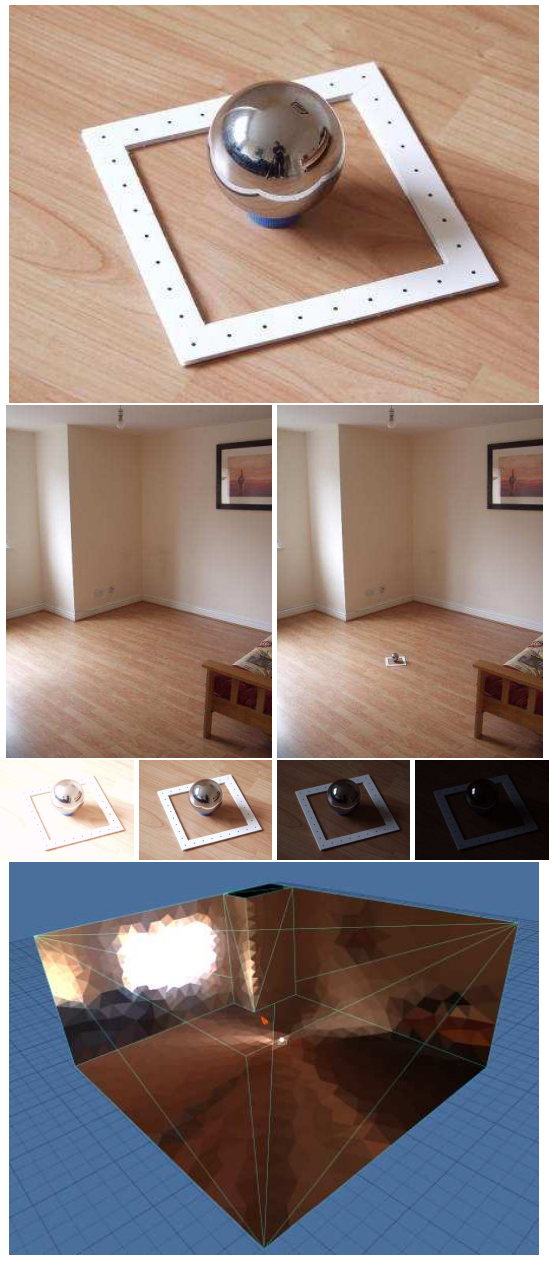


**Figure 7:** *A light probe and calibration grid (top) are positioned in the scene (middle), and HDR radiance data reconstructed and projected outwards from the lightprobe onto the model geometry (bottom).*

## 5. Shading Synthetic Objects

Here we will briefly describe how we use the patch mesh to illuminate synthetic objects. The diffuse component is evaluated using an irradiance volume[34], which is constructed as a pre-process, and is used to encode an approximation of the 5D representation of irradiance (3 positional coordinates, and 2 directional coordinates). The scene is subdivided into a uniform grid, and irradiance is sampled and stored at each grid vertex. The directional distribution of irradiance at each vertex is encoded using spherical harmonic coefficients. It has been shown previously that only 9 coefficients are required to accurately represent irradiance[58, 59], and this enables the irradiance volume to be constructed very rapidly and stored using a small amount of memory. For example, the 20*x*10*x*20 volume built for the scenes in Figure 17 required on average around 30 seconds to construct, and occupied only 0.5 Mb of memory. Typical examples of irradiance volumes are given in Figure 8.

In order to shade each vertex of a synthetic object, we use a simple table lookup into the irradiance volume, using tri-linear interpolation of the spherical harmonic coefficients. The surface normal is then used to retrieve an approximate irradiance value for the vertex. This irradiance is then reflected diffusely using the objects diffuse reflectivity, and the result is tone-mapped using a table look-up into the pre-calculated camera response function. Using this technique, we are able to achieve shading rates of approximately $350,000$ vertices per-second on a 2.5 GHz Pentium 4 CPU, which is equivalent to shading an object with $11,5000$ vertices at over 30 frames-per-second.

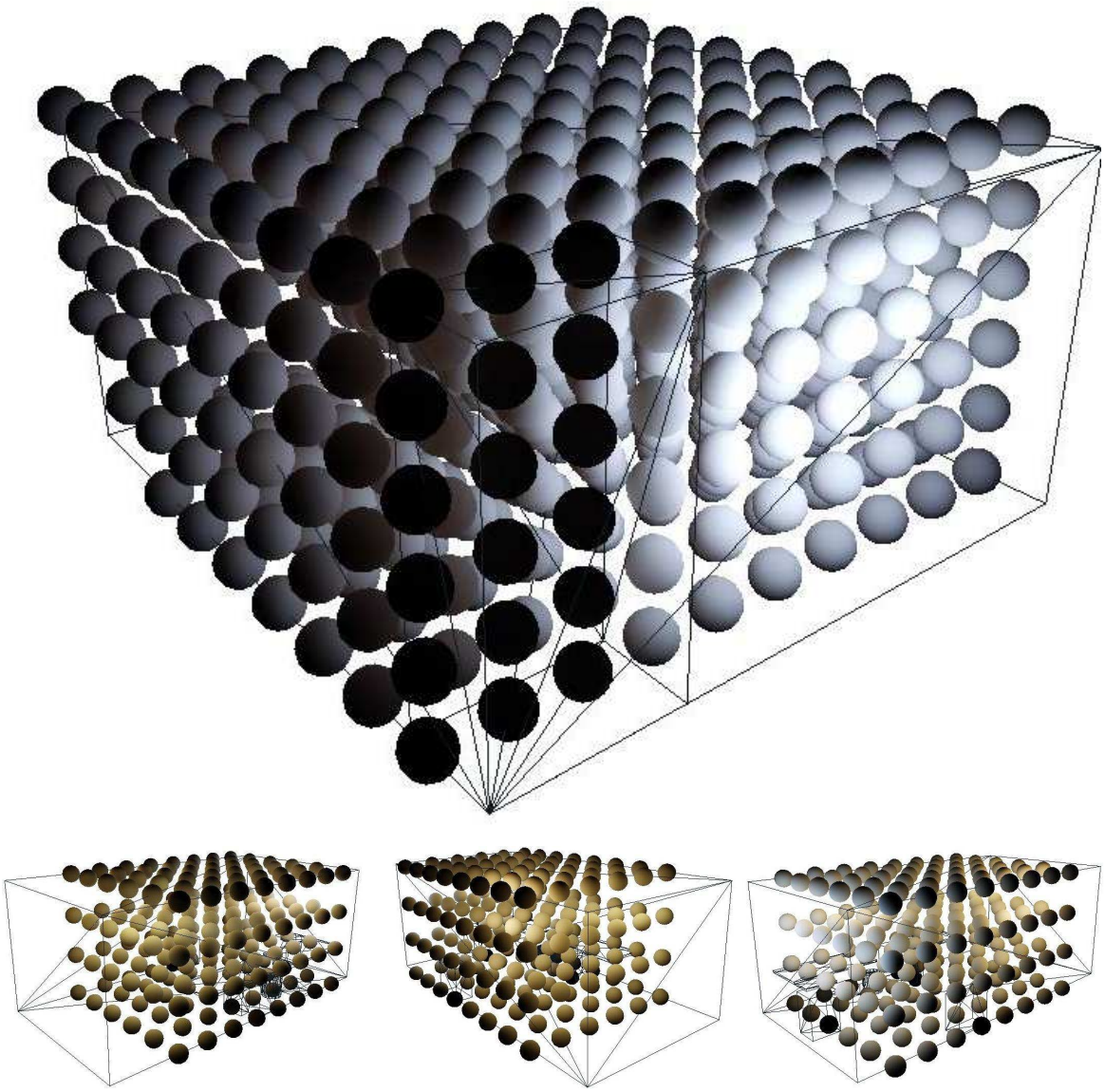


**Figure 8:** *Example irradiance volumes. The irradiance volume for the scene in Figure 7 is shown at the top. The window corresponds to the bright area on the right-hand side, where you can clearly see the increased irradiance at the distance from the window decreases. On the bottom two rows are views of the irradiance volume for the scene shown on the right of Figure 16. This scene is lit by both artificial and natural light, and the difference in colour and intensity of the illumination can clearly be seen within the irradiance volume.*

Specular reflection is evaluated with a separate rendering pass, using a simple Phong-like illumination model and dynamically-generated environment maps. Because of the problem of passing high-dynamic range values through the OpenGL pipeline, we tone-map our scene mesh before generating a low-resolution cubic reflection map for each material in each frame. Radiance values at each mesh vertex are weighted by the specular reflectivity of the material, and then tone-mapped using the camera response function. OpenGL environment-mapping facilities are used to combine the specular component with the diffuse component, which is previously evaluated at each vertex of the object, as described above. Currently, we do not support glossy re-

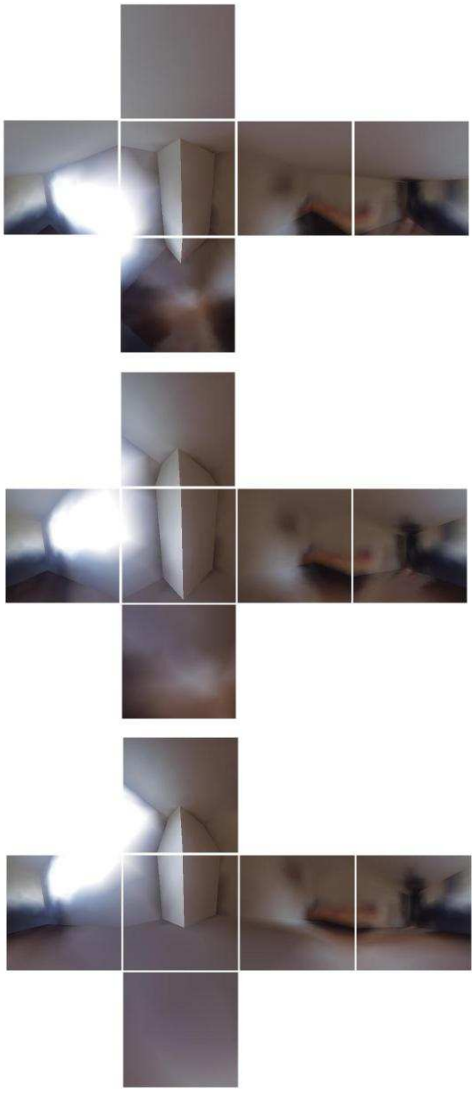


**Figure 9:** *Dynamically generated cubic reflection-maps, taken from the environment shown in Figure 2. Starting near the floor, the environment maps are generated dynamically as the object is raised upwards (bottom to top).*

flection, although this could be approximated by blurring the specular environment maps before rendering.

Although these techniques for shading objects are not entirely accurate, we have found them to be sufficient for generating believable representations of objects illuminated by real-world lighting environments. It is important to note that the shadowing algorithms described above are independent of these shading algorithms. For increased accuracy, more sophisticated algorithms similar to those described in[44, 60, 69, 68] could be used.

## 6. Shadow Generation

It is important to note the assumptions we are making in order to generate soft shadows at interactive rates. Most significantly, we assume that a soft shadow can be accurately represented using multiple overlapping hard-edged shadows[41]. Whilst this is rarely true when using small numbers of hard shadows, we will show that our algorithm is capable of achieving interactive frame-rates whilst using a large number of shadow blending passes, which allows a much wider variety of soft shadows to be approximated. The number of blending passes can be increased or decreased at run-time, and we will show that as the number of blending passes increases, our algorithm is able to converge (in a visual sense) to a result that very similar to that achieved using existing ray-tracing and differential rendering algorithms[17]. We also assume that the only moving objects are the synthetic ones we are introducing, and that casting shadows is the only effect these synthetic objects have on the environment.

We use a shaft-based data structure to provide a hierarchical subdivision of the light transport paths within the reconstructed environment. Shafts are used to link a hierarchy of source patches with a hierarchy of the receiver patches visible in the image, thereby allowing us to quickly determine the sources of light that are potentially occluded by any synthetic objects. We will show how hardware accelerated shadow-mapping may be used to identify the pixels in an image where light from these sources is occluded by synthetic objects. Multiple rendering passes are then performed that blend hard shadows together to approximate the soft shadow cast by the object. We will show how the contributions of light may be easily removed from the background image using facilities commonly found on modern graphics hardware. This results in a rendering algorithm capable of generating complex, visually realistic shadows at interactive frame-rates.

Construction of the line-space subdivision relies on the patches in the environment being partitioned into two sets, containing source and receiver patches respectively. Note that a single patch may be classified as both a source and a receiver, and hence may appear in both sets. Also, we make no distinction between primary and non-primary sources of light, and simply take every patch with non-zero radiance as a potential member of the source set. In discussions below, we refer to any patch with a non-zero radiance as a “source patch”.

The *receiver set* contains all patches that are visible from the point of view of the calibrated image camera. The *source set* contains the patches that are considered to provide significant contributions of light to the image. This set is built by first sorting all patches in decreasing order of radiance. The source set is defined as the first $N$ patches in the sorted list having a total power equal to a user-specified percentage of the total power of all patches. This has the effect of removing very insignificant sources of light from further con-

sideration. The percentage of radiance can be used to trade accuracy against shaft hierarchy traversal time, but typically, a value of around 70% has been found to be satisfactory in all situations we have encountered, as this accounts for all primary and important secondary sources of light.

### 6.1. Radiance Transfer Pre-computation

One important assumption we make during the shadow rendering process is that the background environment remains static. This allows us to pre-compute the radiance transfer from each source patch to the vertices of patches contained in the receiver set. Assuming each source patch emits light diffusely, we calculate the form-factor between each source patch and each receiver vertex[15, 66], multiplied by an estimate of the point-to-patch visibility obtained using ray-casting. Because an approximate reflectivity for the vertex has already been estimated, the radiance transfer from one source patch to each receiver vertex can be found, and stored with the source patch. These radiance transfers will be used during shaft-hierarchy traversal to identify shafts that represent insignificant transfers of light, and also during shadow compositing to remove the contributions of light emitted by occluded sources from the background image. Although this is an $O(n^2)$ operation, radiance transfers can be calculated quickly in practice, due to the small number of receiver vertices and source patches.

### 6.2. Shaft-Hierarchy Construction

Before the shaft-hierarchy can be built, patches in the source and receiver sets must be clustered together into separate hierarchies. Patches in the receiver set are clustered using top-down octree subdivision. Subdivision is halted once a node contains less than a user-specified number of receiver patches. Typically, we build the hierarchy with at most 8 patches in one leaf node, but this number can be increased or decreased to trade accuracy against shaft-hierarchy traversal time. For the source set, it is important that we have fine-grain control over traversal of the source hierarchy (see Section 6.2.1 for further details). Because of this, we cluster patches in the source set using top-down binary KD-tree subdivision, which results in a much deeper hierarchy than with an octree. Subdivision is halted once a node contains a single source patch. For non-leaf nodes in the source hierarchy, the total radiance transfer from all child patches to each receiver vertex is calculated, summed, and stored with the node. This will be used in Section 6.4 when generating shadows from non-leaf positions in the hierarchy.

Once the source and receiver hierarchies are in place, the sets of line segments connecting nodes in the source and receiver hierarchies can be constructed using a hierarchy of shafts[35, 22]. The purpose of the shaft hierarchy is to allow the sources of light that are potentially occluded by an object to be quickly identified.

Shaft-hierarchy construction proceeds in a relatively straightforward manner, starting with a shaft linking the root of the source hierarchy to the root of the receiver hierarchy. At each level the planes bounding the region of line-space between patches in the source and receiver nodes are stored with the shaft. Each shaft is recursively subdivided until the leaves of both the source and receiver hierarchies are reached. For each shaft, the total radiance transfer from its source patches to each of its receiver patch vertices is calculated. Recursion is terminated if it is found that the total radiance contribution from the shaft's source patches to each of its receiver vertices is less than 2% of the total radiance associated with the vertex. This avoids using many shafts to store visually insignificant contributions of light[40], which in turn accelerates traversal of the shaft hierarchy and reduces memory requirements.

The shaft hierarchy introduced in this work has certain similarities to that proposed by Drettakis and Sillion[22]. The main difference between the two approaches is that our hierarchy is only used to store a coarse representation of existing light transport paths in order to identify the source patches that are potentially affected by a moving object. Once these sets of patches have been identified, shadow mapping hardware is used to resolve the fine-grain occlusions of light (see Section 6.3). Because we are encoding an existing static lighting solution, we are also able to remove shafts that transfer insignificant contributions of energy. This is in contrast to the hierarchy proposed by Drettakis and Sillion, which is used to encode the complete set of light transport paths in an environment. As will be demonstrated later, this separation of coarse and fine-level evaluation allows our shaft hierarchy to be constructed very quickly using a small amount of memory (see Section 7).

#### 6.2.1. Hierarchy Traversal

In order to augment an image with shadows cast by a synthetic object, the sources of light occluded by the object must be rapidly identified. The shaft hierarchy described above is used to perform this task, and in this section we outline how a list of potentially occluded source patches may be generated.

Given the bounding box of a synthetic object at one particular instance in time, we are able to quickly identify the set of shafts that intersect this box and are therefore potentially occluded by the object. This traversal of line-space is done by visiting each node of the shaft-hierarchy recursively, starting at the root. An intersection test is applied between the shaft $s$ and the object's bounding box[35]. If the box does not intersect $s$, further traversal of the portion of line-space associated with the shaft can cease. Alternatively, if an intersection occurs, the test is applied recursively to each of $s$'s children. If $s$ is a leaf shaft then the source patch $p$ associated with $s$ is added to a list. $p$ is then tagged with a frame-number counter that is incremented after every frame

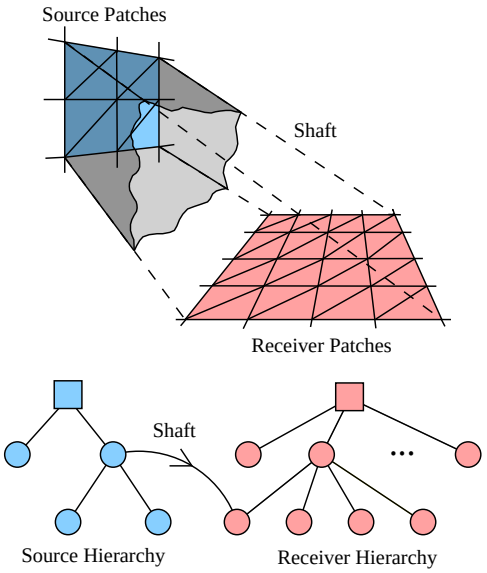



**Figure 10:** *Shaft-based subdivision of the line-space between source patches (blue) and receiver patches (red). A shaft is built which bounds the line-space between each node in the receiver and source patch hierarchies.*

is rendered. As further source patches are found their counter tags are checked against the current frame number to make sure each patch is not added to the list multiple times. Once traversal of the shaft-hierarchy has been completed, we are left with a list of source patches that may cast shadows from the synthetic object (the *source list*). Similarly, by placing the receiver patches associated with the leaf shafts in a *receiver list*, we are also able to identify the regions of the scene that will potentially receive a shadow cast by the synthetic object.

The shadow compositing algorithm described in the next section generates a single hard shadow for each of these source patches, blending them together to form an approximation to the correct shadow. Typically, the time required to do this will exceed the amount of time the user is willing to spend generating each frame. For this reason, our rendering algorithm is able to use the source hierarchy to trade accuracy against frame-rate and generate single hard shadows from groups of source patches in order to render a single frame within the available time. The mechanisms by which this is achieved will be described in Section 6.4.

The memory requirements and time required to traverse line-space depend on the complexity of both the source and receiver hierarchies. As mentioned above, we use an octree subdivision and large leaf size for the receiver hierarchy, and deeper KD-tree subdivision with a small leaf size (i.e. a single patch) for the source hierarchy. The octree subdivision of receivers results in a broad but relatively shallow receiver hierarchy, meaning that large regions of line-space may be quickly removed from consideration, and traversal to the leaf nodes occurs rapidly. For the source hierarchy, however, more fine-grain traversal is required in order to meet the required frame rate. Because subdivision of a binary KD-tree node only increases the total number of leaf nodes by one, this structure is used to store the source patch hierarchy.

### 6.3. Shadow Compositing

The process of compositing shadows into the background image occurs after the synthetic objects have been shaded and depth composited with the scene model. The overall approach we take is to generate a shadow-map for each patch in the source list, and use this shadow-map as a mask to remove the corresponding contribution of light from the background image in regions where the source is occluded from receivers by the synthetic object. This process is repeated for each source patch, blending multiple shadows into the background image and results in a subjectively realistic representation of the real shadow. By using facilities available on modern graphics hardware, the generation of these shadow-maps and the removal of light contributions from the background image can be done quickly enough to allow frames to be generated at interactive rates. In the following discussion we will assume that we are generating a single shadow from each patch in the source list. In Section 6.4 we will show how this assumption may be lifted, allowing the overall rendering speed and quality to be increased or decreased. The algorithm described here is a modification of the differential rendering algorithm introduced by Debevec[17], enabling us to work with standard low dynamic-range frame buffers found in commonly available graphics hardware.

The differential rendering algorithm introduced by Debevec describes how two synthetic images of a scene may be used to compute the changes in a background photograph caused by the introduction of synthetic objects. Given a rendered image $I_{obj}$, containing the synthetic objects and scene geometry illuminated by the reconstructed lighting data, and a second image $I_{noobj}$ that does not contain the synthetic objects, the difference between these two images, $I_\varepsilon$, is subtracted from the background photograph $I_b$:

$$I_{final} = I_b - I_\varepsilon = I_b - (I_{noobj} - I_{obj}) \qquad (2)$$

in order to generate a final image $I_{final}$ that contains the correct shadowing effects. Wherever $I_{obj}$ is darker than $I_{noobj}$ (i.e. the areas where the synthetic object cast a shadow), light is subtracted from the background image accordingly.

More specifically, consider a pixel in the image, and a point $x$ which corresponds to the nearest surface seen through that pixel. The adjustment $\varepsilon_x$ that must be subtracted from the radiance associated with the pixel is simply:

$$\varepsilon_x = \sum_{j=0}^{N-1} L_{xj} - \sum_{j=0}^{N-1} L_{xj} V_{xj} = \sum_{j=0}^{N-1} L_{xj} M_{xj} \qquad (3)$$

1. Pre-process:
   For each source patch $j$
      For each receiver vertex $i$
         Calculate $L_{ij}$

2. Repeat for each frame:
   Render the background image
   Render the synthetic objects
   For each source patch $j$
      Enable shadow mapping to multiply by $M_{ij}$
      Subtract contribution from $j$ from the frame-buffer by rendering the receiver mesh with vertex colours set to $L_{ij}$

**Figure 11:** *Two stage compositing process for differential shadow rendering.*

where the summation is over all source patches $j = 0 \ldots N-1$, $L_{xj}$ is the unoccluded radiance transferred from source $j$ to $x$ and then reflected at $x$ towards the camera, and $V_{xj}$ is the visibility of $j$ with respect to $x$, i.e. $0 \leq V_{xj} \leq 1$, where $V_{xj} = 0$ if the transfer is completely occluded by a synthetic object, and 1 if it is completely visible. Defining a new term, $M_{xj} = 1 - V_{xj}$, allows the adjustment to be calculated using a single summation, where $M_{xj}$ represents an *occlusion mask*, which varies between 1 when $i$ is completely occluded from $j$ , and 0 when it is completely visible.

In order to apply these adjustments to a background image, we assume that the background scene is static, and separate the term inside the summation in Equation 3 into two parts: $L_{xj}$ which can be pre-computed for each $x$ and $j$, and $M_{xj}$ which depends on the position of the dynamic synthetic objects.

In order to execute this algorithm at rates fast enough for interactive applications, we take the basic approach of performing the image generation and subtraction operations in Equation 2 using graphics hardware. In the following discussion, we assume that the graphics hardware and frame-buffer are able to process HDR data. Once the basic algorithm is described, extensions that allow us to work with low dynamic-range (LDR) data will be presented in Section 6.3.1. Facilities to perform these LDR operations are available on NVIDIA GeForce3/4 graphics hardware, using extensions to OpenGL 1.2.

We first assume that the contribution of a single source patch $j$ to each scene point $x$ is smoothly varying, allowing us to store $L_{ij}$ for each $j$ at the vertices $i$ of patches in the receiver set. We let the graphics hardware linearly interpolate the values between each receiver vertex. Differential rendering of shadows into a background image can then be performed using the two-stage process presented in Figure 11. Note that we have explicitly separated the calculation of $M_{ij}$ from the subtraction of $L_{ij}$. This is done because of the different rendering techniques are used to execute each loop: The first is evaluated using hardware shadow-mapping, approximating $M_{ij}$ at each pixel in the image using binary visible/invisible values. Subtractive blending is then used during the second loop, and the receiver set is drawn with the colour of each vertex $i$ set to $L_{ij}$. Texture combiners are set to use the shadow-map as a mask, simulating the multiplication by $M_{ij}$.

### 6.3.1. Shadow Compositing using Graphics Hardware

The discussion so far has only considered HDR representations of light where, assuming access to a floating-point frame buffer, we can operate entirely on floating-point radiance values and map back to pixel intensities as a post-process. Complications occur, however, when we try to apply differential rendering algorithms to LDR images, as used by most digital cameras and graphics hardware. Most importantly, for the background image we wish to augment, the relationship between high and low dynamic-range representations of light is non-linear. Ideally we would like to perform all operations using HDR data and apply a non-linear tone-map after shadow compositing:

$$I_{final} = T(L_{final}) = T(L_b - L_\varepsilon)$$

where $I = T(L)$ is the tone-map transforming radiance into pixel colours. Unfortunately, due to the LDR nature of the frame-buffer we must operate entirely with LDR data.

By letting the graphics hardware interpolate between vertices in the receiver set, we can reduce the problem to one of performing differential rendering at the receiver vertices themselves. We will denote the desired HDR differential rendering process at a vertex $i$ as:

$$I_{final_i} = T(L_i - \sum_{j=0}^{N-1} L_{ij} M_{ij})$$

where $L_i$ represents the radiance obtained from the image at the pixel location associated with vertex $i$. Define a new *intensity transfer* $S_{ij}$ for each pair of a vertex $i$ and source patch $j$. These intensity transfers are LDR equivalents of the radiance transfers $L_{ij}$ in Equation 3. We wish to subtract these intensity transfers from the LDR frame-buffer intensity $I_i$ so that the overall result is equivalent to when HDR operations are used:

$$I_{final_i} = I_i - \sum_{j=0}^{N-1} S_{ij} M_{ij} \tag{4}$$

Because we will be removing these contributions from the frame-buffer using multiple rendering passes, and we do not know the correct values for $M_{ij}$, Equation 4 implies that:

$$S_{ik} = I_i - T(L_i - \sum_{j=0}^{k} L_{ij} M_{ij}) - \sum_{j=0}^{k-1} S_{ij} M_{ij} \tag{5}$$

must hold for each $0 \leq k < N$. Unfortunately, we are unable to pre-compute the intensity transfers exactly from this

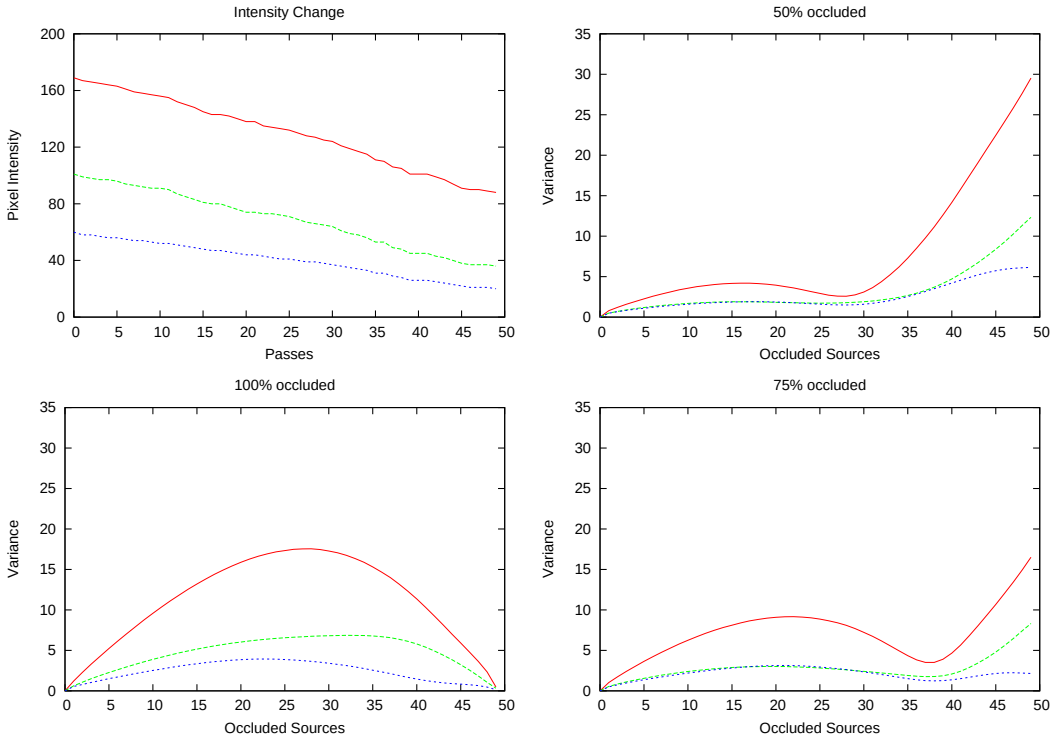



**Figure 12:** *The reduction in frame-buffer intensity as increasing number of shadow passes are applied (top left), and the error (in pixel colour) caused by the assumption that all patches are occluded in two equally-sized sets (top right). Errors for two sets of different sizes are shown on the bottom row.*

relation, because the values of $M_{ij}$ are not known until rendering occurs. The non-linearity of $T()$ also means that the final result is dependent on the order the source patches are considered. We can, however, generate a useful approximation by assuming that each source patch is either entirely visible or entirely invisible. Initially, we don't know which of the source patches will be visible and which will be invisible, but if we assign estimates to each source patch then we can calculate $S_{ij}$ and remove the correct contribution from the background image. If the visibility estimates were correct, this should result in a correct final image, assuming the order that the source patches are considered remains the same. In practice, the order is unlikely to remain fixed, but if we choose to order patches from brightest to dimmest when evaluating Equation 4, and ensure we sort any later sets of source patches in that same order, the approximation error will be reduced.

Without knowing which patches are actually occluded, we can generate an approximation by randomly partitioning the source patches into two separate sets. By assuming that when all the patches in the first set are occluded those in the second remain visible, we can fix the values of $M_{ij}$ and calculate intensity transfers for the first set of patches using Equation 5. Similarly, assuming that when the patches in the second set are all occluded, those in the first set are visible, we can determine the remaining intensity transfers.

Figure 12 illustrates how this approximation affects the final shadow intensity for differently sized sets. The graph in the top-left shows the typical reduction in $I_i$ that occurs after each successive rendering pass using a set of 50 random source patches. The remaining graphs plot the error found when assuming that all source patches are occluded in differently sized sets. Intensity transfers were calculated as described above. Varying numbers of $P$ ($0 \leq P \leq 50$) source patches were then randomly selected as being *actually occluded*, simulating the evaluation of $M_{ij}$ using shadow-mapping (plotted on the horizontal axis of each graph). For each $P$, 10000 trials were run over 4 datasets, and $P$ random patches were selected for each trial. The difference between the left and right-hand sides of Equation 4 was then measured, with $M_{\cdot j} = 1$ for the $P$ random patches, and 0 otherwise. The graph shows the variance of the error in red green and blue pixel intensities.

For each set size, the error is insignificant for small $P$. This is because subtracting a small number of incorrect intensity transfers has little effect on the overall image. Similarly, the

1. Pre-process:
   Sort source patches in decreasing order of transfer

2. For each receiver vertex $i$:
   $V_1 = V_2 = L_i$
   $C_1 = C_2 = T(L_i)$
   For each contributing source patch $j$:
      if $j$ is even
         $V_1 = V_1 - L_{ij}$
         $C_1' = T(V_1)$
         $S_{ij} = C_1 - C_1'$
         $C_1 = C_1'$
      else
         $V_2 = V_2 - L_{ij}$
         $C_2' = T(V_2)$
         $S_{ij} = C_2 - C_2'$
         $C_2 = C_2'$
      endif

**Figure 13:** *Pseudo-code for estimating intensity transfers, executed before drawing each frame.*

error is also small for values of $P$ that match the assumption being made (e.g. the error is small for $P = 25$ when assuming an 50%/50% split). For intermediate values of $P$, the error rises as increasing numbers of incorrect intensity transfers are subtracted from the image.

In practice, we have found that for receivers in the vicinity of synthetic objects, typical occlusion rates run at around $30-50\%$ for the scenes we have examined, and only rarely rise above 75%. For this reason we have used the 50%/50% split in all further examples because this split has the smallest overall error in the $30-50\%$ region (see the top-right graph in Figure 12).

### 6.3.2. Calculating Intensity Transfers

Intensity transfers can be calculated very quickly for each frame before the shadows are composited into the background image. Before these intensity transfers can be determined, the patches in the source list for the current frame are sorted in decreasing order of average radiance transfer to patches in the receiver set. The average transfer of radiance from each source patch can easily be pre-computed and stored with the source hierarchy because we assume that light reflected off the synthetic objects does not affect the overall illumination in the scene. The transfers can then be calculated using the algorithm presented in Figure 13. For each receiver vertex, $V_1$ and $V_2$ are initialised to the total radiance gathered from all source patches and reflected at the vertex towards the camera. These two radiance values will be used to calculate the intensity transfers under the assumption that the source patches are occluded in two equally sized sets, as described above. These initial radiance values are mapped to pixel colours $C_1$ and $C_2$ using the calibrated camera response function $T()$.

A loop is then made over all patches in the source list that can contribute radiance to the vertex. In order to quickly simulate a random assignment of patches to sets, we assign each patch according to a randomly generated id number between 0 and $N-1$. For even numbered ids, the pre-calculated radiance transfer from the source to the receiver is subtracted from $V_1$, and the radiance is then transformed by $T()$ into a pixel colour $C_1'$. The intensity transfer $S_{ij}$ is then calculated as the difference between $C_1$ and $C_1'$. $C_1$ is set equal to $C_1'$ and the process repeated for the next source patch. For odd numbered ids the calculations are performed using $V_2$ and $C_2$, so as the source list is traversed, two independent radiance values are used to estimate the intensity transfers. Each of these independent values corresponds to one of the sets we made in the occlusion assumption described above.

### 6.3.3. Shadow-Map Generation

As a pre-process, simplified representations of all synthetic objects are generated using the techniques described in[26], each containing between 100 and 500 triangles. These simplified objects are used during shadow-map rendering, and shadow-map resolution is also limited to 256x256 pixels. This greatly accelerates rendering speed without visibly reducing image quality.

Once the intensity transfers have been estimated for the current frame, the second inner-loop of the algorithm presented in Figure 11 can be executed, with $L_{ij}$ replaced by the transfers $S_{ij}$. The receiver set is drawn with vertex colours set to $S_{ij}$, and graphics hardware used to interpolate between these values. A shadow map is then generated for each source $j$, allowing us to find $M_{\cdot j}$. This is done by first initialising the OpenGL projection and model-view matrices so the synthetic object is contained entirely within the shadow-map, as seen from the source patch. The simplified representation of the synthetic object is then rendered into the depth buffer to produce the shadow-map. Hardware shadow-mapping, texture combiners, and blending operations are initialised so that when the geometric representation of the receiver set is drawn, the vertex colours ($S_{ij}$) are multiplied by $M_{\cdot j}$, and the product is subtracted from the background colour buffer. If required, self-shadows cast onto the synthetic objects can also be generated by approximating the intensity transfer $S_{ij}$ from a source patch to the vertices of the object, and then rendering the object with shadow-mapping and blending enabled.

## 6.4. Controlling Frame-Rate

In the previous section we described how a shadow from each source patch could be generated and composited into a background photograph using commonly available graphics hardware. In interactive settings, the time required to do

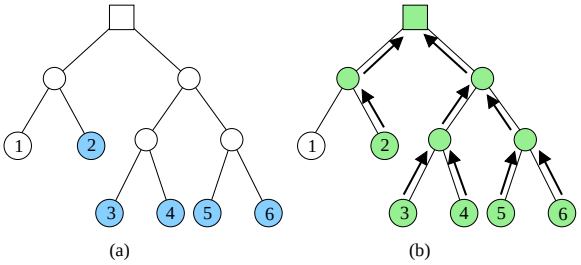



**Figure 14:** *(a) Traversal of the shaft-hierarchy identifies* 5 *out of* 6 *potentially occluded source patches (marked in blue). (b) The affected portions of the source hierarchy (shown in green)are then identified by pushing a frame identification tag up towards the root node.*

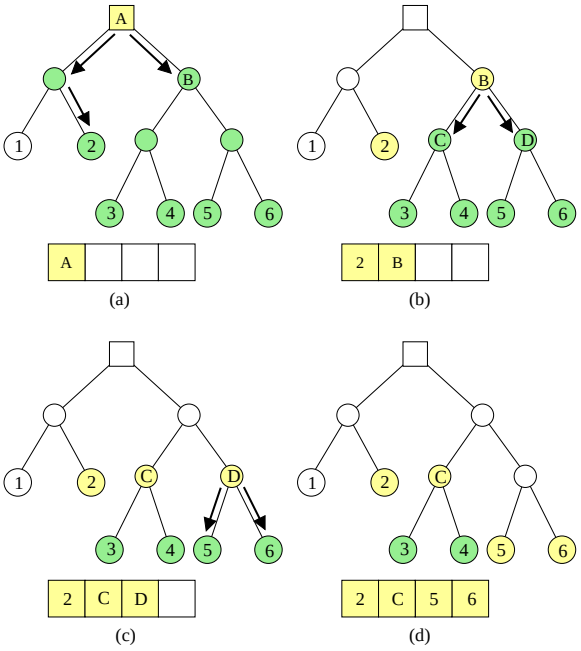



**Figure 15:** *A set of* $n = 4$ *source clusters are identified that represent the combined effect of all* 6 *potentially occluded source patches. Starting at the root node (a), the affected portion of the hierarchy is traversed (b), (c), and (d) until the required number of source clusters are identified (yellow).*

this for all source patches will often exceed the time a user is willing to spend generating a single frame. In these situations, what is required is a trade-off between overall shadow quality and rendering cost, and this can be achieved by generating shadow-maps from non-leaf nodes in the source hierarchy that was described in Section 6.2.

Figure 14(a) shows a typical source hierarchy, with a root node at the top and 6 source patches at the leaves. Assume, for example, that during construction of the source set, $N = 5$ out of the 6 source patches (shown in blue) have been identified as being potentially occluded by a synthetic object. Assuming that we are unable to generate shadows from all 5 sources due to frame-rate constraints, we need to find a *representative source set* that encapsulates the effect of all source patches and yet can be processed in the available time. We can do this very quickly before each frame is rendered using the algorithm described in this section.

Building the representative source set starts by pushing the current frame number from each potentially occluded source patch up towards the root of the hierarchy. This allows the branches of the hierarchy containing these patches to be identified and marked (shown in green in Figure 14(b)). Starting with the root, we wish to build a list of $n < N$ nodes, where each node can be either a leaf of the source hierarchy or an intermediate node representing the combined effect of several leaves.

Figure 15(a) shows the start of the construction process for $n = 4$ nodes. While the target number of nodes has not been reached, the node in the list which transfers the largest average amount of radiance to patches in the receiver set is removed from the list. This is done very quickly by storing the list using a binary tree, sorted by the average radiance transfer. Initially, as it is the only node in the list, the root node is removed. The hierarchy is then traversed by one level and the node's immediate children in the marked portions of the hierarchy are identified and added to the list (source patch "2" and node "B"). This process is repeated until the required number of patches or nodes is found. Figures 15(b) and (c) show further traversals, until finally, in (d) we reach the target of $n = 4$ nodes. Note that although we have chosen fewer than 5 nodes, the energy from all potentially occluded source patches is still accounted for, because the radiance transfer from node "C" represents the combined effect of source patches "3" and "4". When rendering a single shadow-map from a cluster of sources, such as node "C", the origin of the source is chosen to coincide with the centre of the patch that contributes the most energy to the receivers.

## 7. Example Augmented Environments

The algorithms described here have been implemented using OpenGL on a 2.5 GHz Pentium 4 PC running Microsoft Windows XP and equipped with a NVIDIA GeForce 4 Ti4600 graphics card. Examples showing interactive object movement are given in Figure 16. All images are snapshots from an interactive session rendered at approximately 15 frames-per-second, using 50 blending passes. Of this, the majority of the time was spent generating and blending shadows into the background image, and the time required to shade each object was negligible. The left-hand column shows an example where the user is lifting a box off the floor of the scene. Because the intensity of the shadows blended into the background image are based on the actual amount of light transferred from source to receiver, the reduction in

**Figure 16:** *Examples of interaction between a synthetic object and real environment, generated at around 15 frames-per-second using our system. The left-hand column illustrates the reduction in shadow intensity that occurs as the synthetic object is raised off the ground. The sequence in the right-hand column shows how real and virtual shadows can interact as the synthetic object is moved underneath the real desk.*

intensity of the synthetic shadow is correctly modelled as the object is raised off the floor. The right-hand column shows a different kind of interaction between a synthetic object and the environment, where the sphere moves under a real table. Notice the darkening of the object and merging of the synthetic shadow with the real shadow due to the fact that the desk has been included in the geometric scene model. Further examples are given in the accompanying video material.

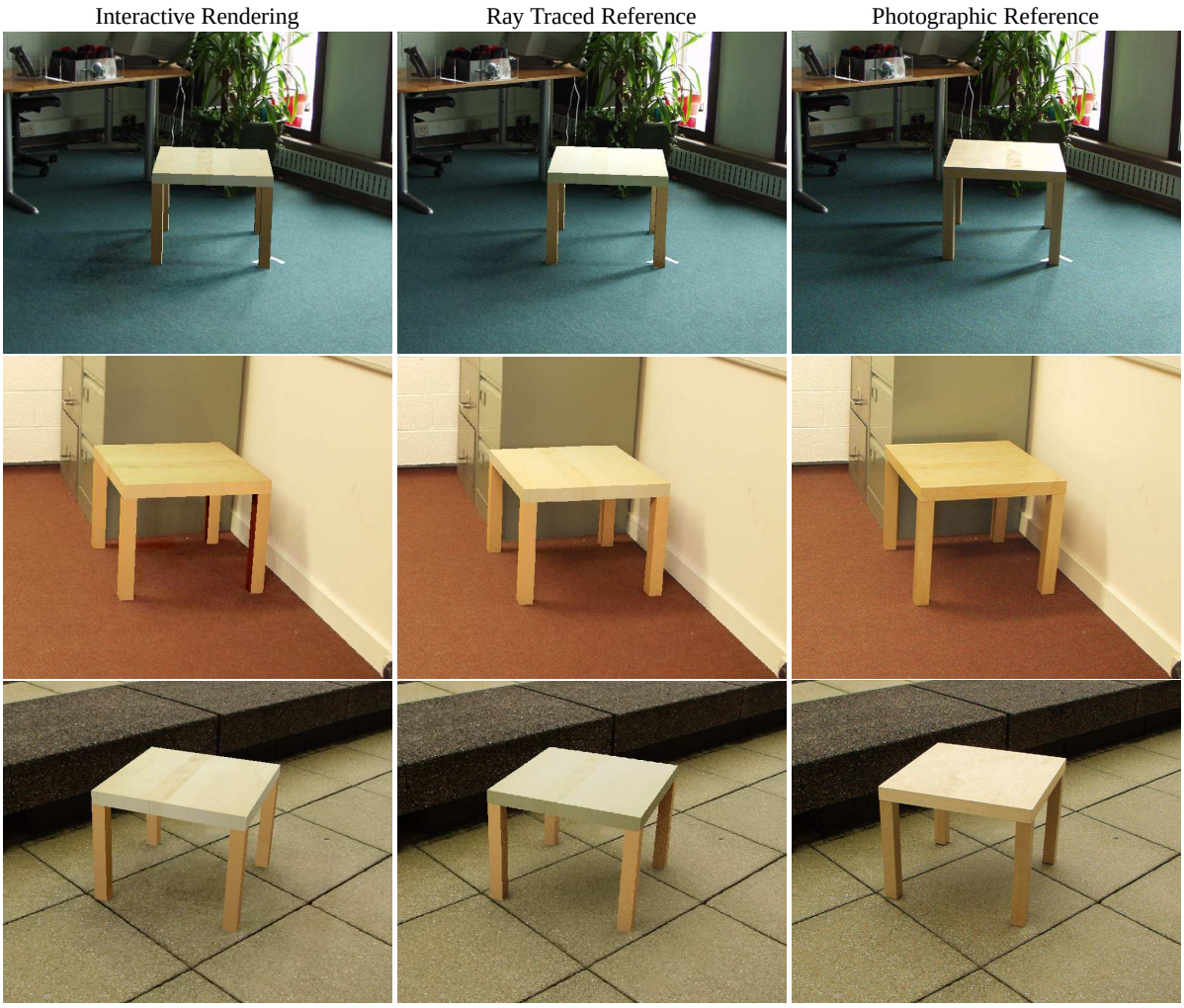



**Figure 17:** *A comparison of image quality for three different scenes, containing both soft and harder-edged shadows cast by daylight and artificial light sources. Snapshots from interactive sessions with our system are shown on the left, generated at (from top to bottom) 14, 12 and 16 frames-per-second respectively. Ray-traced reference images are shown in the middle column, and photographic references containing an equivalent real object at approximately the same position are shown on the right. Further details are given in the text.*

Figure 17 compares rendering quality against ray-traced and photographic references for different lighting environments. In each row, an image produced using our interactive system is shown on the left, a ray-traced image generated using a HDR differential rendering algorithm[17] is shown in the middle, and a photograph of a real object in the scene is shown on the right. The left-hand images were all generated using 50 blending passes, and were rendered at 14, 12 and 16 frames-per-second respectively. For comparison, the ray-traced images in the middle column each took several hours to generate using an un-optimized Monte-Carlo ray-tracer. Overall, the shadows generated using our algorithm are subjectively very similar to both the ray-traced and photographic references.

Note that the differences in shading of the synthetic objects in these examples are due to the fact that we did not accurately measure or model the reflectance properties of the real object. As such, the shading is only an approximation and these images are only intended to indicate the quality of the shadows that our rendering algorithm can generate.

The time spent constructing the patch and shaft hierarchies for these examples was relatively small. The first example in Figure 17 required around 30 seconds of pre-processing time, and produced a shaft hierarchy with $47,000$

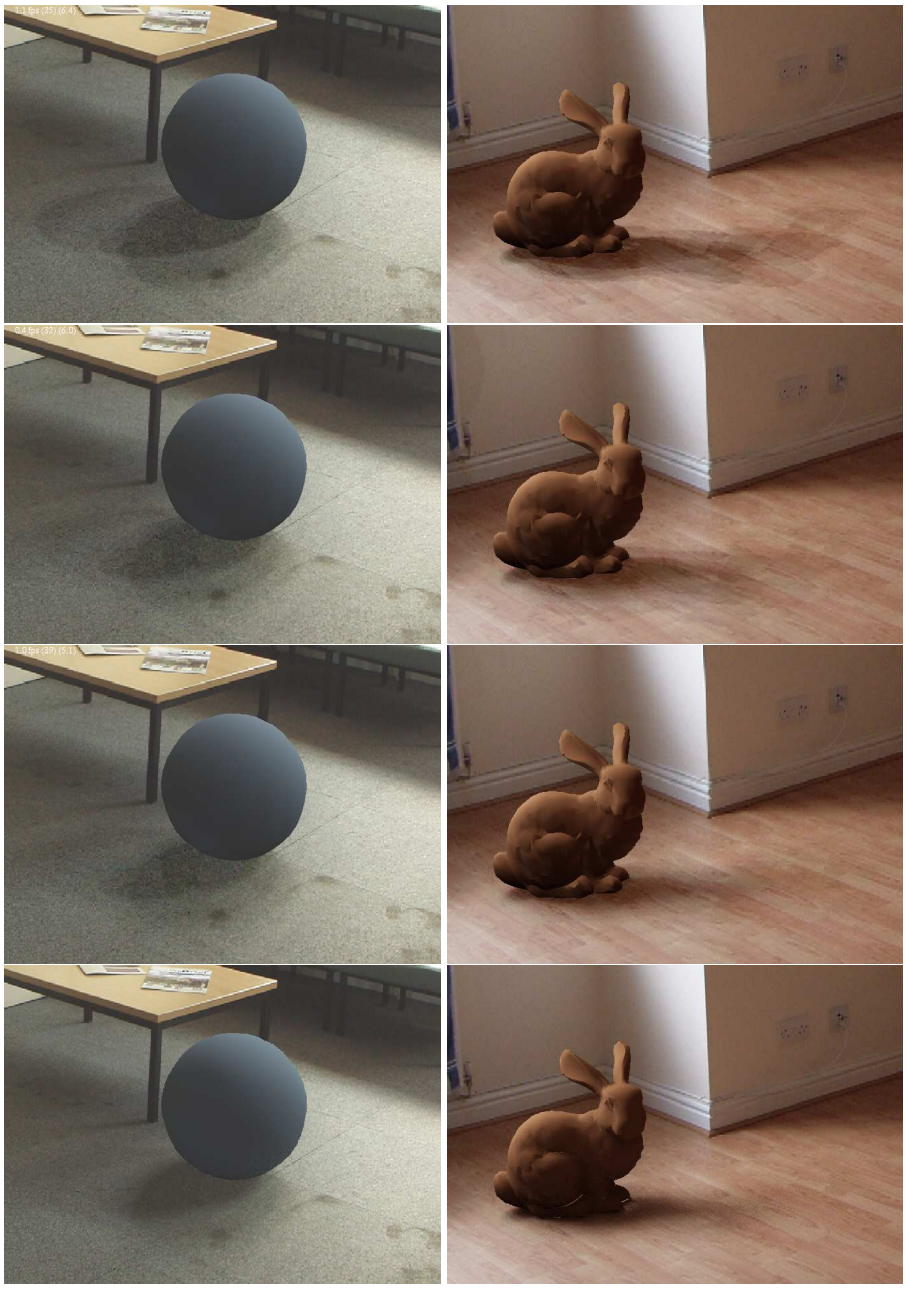


**Figure 18:** *This figure shows the trade-off between rendering speed and accuracy that is available with our system, for two different lighting environments. From top to bottom, this figure presents snapshots from our system with shadows generated using 10, 20 and 50 blending passes. These images were rendered at approximately 35, 25 and 14 frames per second for the left column, and 27, 18 and 9 frames per second for the right column. For comparison, ray-traced reference images are shown on the bottom row.*

leaf nodes occupying just over 9 Mb of memory. The second and third examples only required 15 seconds of preprocessing, generating 102,000 and 64,000 leaf shafts respectively. Typically, each scene contained between 1000 and 2000 patches, and between 10 and 40 milliseconds was required to traverse the shaft hierarchy and generate a representative source set for each frame.

The trade-off that can be made between frame rendering time and shadow accuracy is illustrated under two different lighting environments in Figure 18. On the top row, a 500 triangle sphere was rendered into a background environment using different numbers of blending passes. For each number of passes, the algorithm described in Section 6.4 was used to determine a representative set of source nodes, and a single shadow-map was generated for each node. From left to right, the Figure shows frames generated with 10, 20 and 50 passes. These were rendered at rates of 35, 25 and 14 frames-per-second. For comparison, a ray-traced image was also produced in approximately 1 hour and is shown on the right. Note that because the image on the far-left was generated with a smaller number of blending passes, the hard-edged shadows are clearly visible. Our algorithm is, however, able to maintain the same overall intensity of the shadow as in the ray-traced image (see Section 6.3.1). As the number of blending passes increases, the hard edges of the individual shadow-maps are less visible and the overall result becomes an increasingly better approximation to the ray-traced reference image on the far-right. Similar images for a second lighting environment are given on the bottom row. Frame rendering rates for this example were 27, 18 and 9 frames-per-second respectively, with the ray-traced image requiring over 1.5 hours to render.

## 8. Perception and Realism

Realness - the state of being actual or real. Obviously this definition refers to the 'real' world and our perception of it, however frequently in the doctrine of computer science the terms 'realistic', 'realism' and 'real' are discussed. Obviously anything represented on a computer is not real but just an approximation, so what do these expressions refer to?

There are many uses for computers in the world we live in ranging from high performance games to high accuracy mathematical calculations. Both of these examples and countless more have one thing in common the need to have some level of realism. Within the games industry it is important for there be some link with reality (or at least some conceivable fantasy of reality) to involve the player in the game. However the level of realism needed in a computer game is related to the genre and objective of the game. At the other end of the spectrum there applications that directly apply to the real world; one example is the package that performs the aerodynamics calculations that go into producing a new fighter aircraft. In this circumstance an extremely high approximation of reality is needed to ensure that the plane will fly. So within a computer game it is important that the plane looks realistic and behaves like we would expect, however in the mathematical model the appearance of the plane does not matter (and probably isn't even calculated) but it is crucial that the behaviour is as real as possible.

This section is concerned with some of the issues present when attempting to create realistic computer graphics. We start by discussing vision and how we actually perceive the world we live in. Later we move on to discuss several methods of measuring the 'realism' of a computer generated scene.

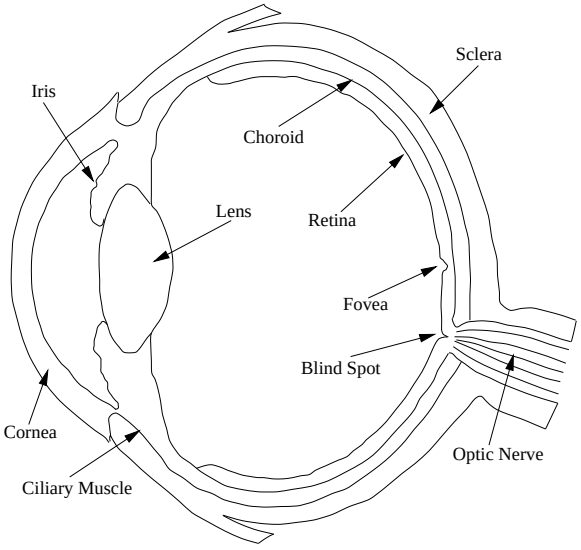



**Figure 19:** *Cross-section of the human eye.*

### 8.1. Visual Perception

The world in which we live comprises of an unfathomable amount of information, fortunately many birds and mammals possess a bewildering array of visual systems which have evolved for the purposes of detecting and using information from reflected light. These range from simple photoreceptors to distinguish light from darkness, to complex chains of actions which lead to cognitive perception. In the case of vision, light in the form of electromagnetic radiation, activates receptor cells in the eye triggering signals to the brain. These signals are not understood as pure energy, rather, perception allows them to be interpreted as objects, events, people and situations. Because of this ability to identify items, visual information has become crucial to many animals for locating and identifying food, suitable habitats, and predators, as well as functioning to orient animals in their overall surroundings.

#### 8.1.1. The Human Visual System

A combination of physics and chemistry within the eye and cognition and perception in the brain gives rise to vision. Reflected light rays enter the eye and are transformed into electrical signals. Brain cells then decode the signals providing us with sight. A diagram of the anatomical components of the human eye is shown in Figure 19. The main structures are the iris, lens, pupil, cornea, retina, vitreous humor, optic

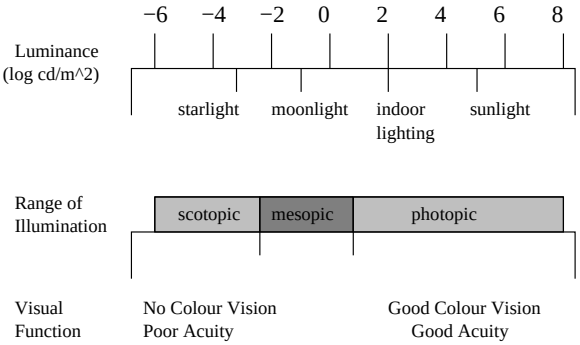



**Figure 20:** *The range of luminances in the natural environment and associated visual parameters, after Ferwerda et al.*[23].

disk and optic nerve. When light enters the eye, it first passes through the cornea, then the aqueous humor, lens (where it is focused) and vitreous humor.

Ultimately it reaches the retina, which is the light-sensing structure of the eye. The retina is a mesh of photoreceptors, which receive light and pass the stimulus on to the brain. The retina contains two types of cells, called rods and cones. Rods handle vision in low light, and cones handle colour vision and detail. When light contacts these two types of cells, a series of complex chemical reactions occurs. The chemical that is formed (activated rhodopsin) creates electrical impulses in the optic nerve. Generally, the outer segments of rods are long and thin, whereas the outer segment of cones are more cone-shaped.

There are millions of cone cells densely packed into the macula. These allow the highly detailed vision needed to read an eye chart or car number plate, or study the stars through a telescope. Conversely rods provided some peripheral vision but are primarily for night vision. Since the rods do not distinguish colour, vision in dim light is almost colourless. Cones provide both luminance and colour vision in daylight. Cones contain three different pigments, which respond either to blue (445nm), green (535nm), or red (570nm) wavelengths of light.

Normal daytime vision, where the cones predominate visual processing is termed photopic, whereas low light levels where the rods are principally responsible for perception is termed scotopic vision. When both rods and cones are equally involved then vision is termed mesopic. Figure 20 shows how this range of luminance is encountered in a natural environment.

### 8.1.2. Visual Acuity and Contrast Sensitivity

This is the ability of the Human Visual System to resolve detail in an image. The human eye is less sensitive to gradual and sudden changes in brightness in the image plane but has higher sensitivity to intermediate changes. Visual field indicates the ability of each eye to perceive objects over a certain spatial range of vision. A normal field of vision is approximately 180 degrees. The standard definition of normal visual acuity (20/20 vision) is the ability to resolve a spatial pattern separated by a visual angle of one minute of arc. Since one degree contains sixty minutes, a visual angle of one minute of arc is 1/60 of a degree. The spatial resolution limit is derived from the fact that one degree of a scene is projected across 288 micrometers of the retina by the eye's lens.

In this 288 micrometers dimension, there are 120 colour sensing cone cells packed. Thus, if more than 120 alternating white and black lines are crowded side-by-side in a single degree of viewing space, they will appear as a single grey mass to the human eye. With a little trigonometry it is possible to calculate the resolution of the eye at a specific distance away from the lens of the eye.

Contrast can be defined as by:

$$(l_{max} - l_{min})/(l_{max} + l_{min}),$$

where $l$ refers to a range luminance values. Contrast sensitivity is a measure of how faded or washed out an image can be before it become indistinguishable from a uniform field. It has been experimentally determined that the minimum discernible difference in grey scale level that the eye can detect is about 2% of full brightness. Contrast sensitivity is a function of the size or spatial frequency of the features in the image. However, this is not a direct relationship as larger objects are not always easier to see than smaller objects.

Human brightness sensitivity is logarithmic, so it follows that for the same perception, higher brightness requires higher contrast. Apparent brightness is dependent on background brightness. This phenomenon, termed conditional contrast is illustrated in Figure 21. Despite the fact that all centre squares are the same brightness, they are perceived as different due to the different background brightness.

### 8.1.3. Depth Perception

Stereo vision gives the ability to see the world in three dimensions and to perceive distance. Various cues allow for depth perception. The difference between the images projected onto the left and right eyes, known as the binocular disparity, is used to discern depth at a close range. Monocular cues are cues to depth which are effective when viewed with only one eye; these include interposition, atmospheric perspective, texture gradient, linear perspective, size cues, height cues and motion parallax.

### 8.1.4. Perceptual Constancy

This is an effect where objects are perceivable alike despite changes in lighting and hence physical vision. A number of perceptual constancies including lightness constancy, colour constancy and shape constancy have been identified by psychologists.

**Figure 21:** *Simultaneous Contrast: The internal squares all have the same luminance, but changes in luminance in the surrounding areas change the perceived luminance of the internal squares.*

- Lightness Constancy: This is the ability to correctly perceive surface lightness despite changes in the level of illumination
- Colour Constancy: This is the ability to correctly perceive the colour of an object despite changes in illumination.
- Shape Constancy: The ability to correctly perceive shape regardless of changes in orientation.

#### 8.1.5. Lightness Perception

Gilchrist[31] defines a lightness error as "any difference between the actual reflectance of a target surface and the reflectance of the matching chip selected from a Munsell chart". An observer can be asked to match the reflectance of simulated objects (in a computer generated rendition of the real world) to the same Munsell chart. This gives a measure of lightness errors with respect to the computer image. Because of limitations of the HVS, errors will be perceived in both the real world and any representation of it. If these sets of errors are similar the representation of the real world can be deemed perceptual similar (at least in terms of lightness) to the real world. This is a relatively simple process to perform making it suitable as a measure of perception (or reality) for some simple scenes. Psychologists have proven that lightness constancy depends on the successful perception of lighting and the 3D structure of a scene. As the key features of any scene are illumination, geometry and depth, the task of lightness matching encapsulates all three key characteristics into one task. McNamara et al.[47] demonstrated this within a simple experimental framework by using measurement of lightness error to calculate optimal rendering parameters for simple greyscale scenes.

### 8.2. Image Quality Metrics

Reliable image quality assessments are necessary for the evaluation of realistic images synthesis algorithms. Typically the quality of the image synthesis method is evaluated using image-to-image comparisons. Often comparisons are made with a photograph of the scene that the image depicts. Several image fidelity metrics have been developed whose goals are to predict the amount of differences that would be visible to a human observer. It is well established that simple approaches like mean squared error do not provide meaningful measures of image fidelity, thus more sophisticated measures which incorporate a representation of the HVS are needed. It is generally recognised that more meaningful measures of image quality are obtained using techniques based on visual (and therefore subjective) assessment of images, after all most final uses of computer generated images will be viewed by human observers.

#### 8.2.1. Perceptually Based Image Quality Metrics

A number of experimental studies have demonstrated many features of how the HVS works. However, problems arise when trying to generalise these results for use in computer graphics. This is because, often, experiments are conducted under limited laboratory conditions and are typically designed to explore a single dimension of the HVS. Instead of reusing information from these previous psychophysical experiments, new experiments are needed which examine the HVS as a whole rather than trying to probe individual components. Using validated image models that predict image fidelity, programmers can work toward achieving greater efficiencies in the knowledge that resulting images will still be faithful visual representations. Also in situations where time or resources are limited and fidelity must be traded off against performance, perceptually based error metrics could be used to provide insights into where corners could be cut with least visual impact.

Using a simple five sided cube as their test environment Meyer et al.[48] presented an approach to image synthesis comprising separate physical and perceptual modules. They chose diffusely reflecting materials to build a physical test model. Each module was verified using experimental techniques. The test environment was placed in a small dark room. Radiometric values predicted using a radiosity lighting simulation were compared to physical measurements of the radiant flux density in the real scene. Results showed that irradiation was greatest near the centre of the open side of the cube. This area provided the best view of the light source and other walls. In summary, there was a good agreement

between the radiometric measurements and the predictions of the lighting model.

Rushmeier et al.[63] explored using perceptually based metrics, based on image appearance, to compare image quality to a captured image of the scene being represented. The goal of this work was to obtain results by comparing two images using models that give a large error when differences exist between images. The following models attempt to model effects present in the HVS. Each uses a different Contrast Sensitivity Function (CSF) to model the sensitivity to spatial frequencies.

Model 1 After Mannos and Sakrison: First, all the luminance values are normalised by the mean luminance. The non linearity in perception is accounted for by taking the cubed root of each normalised luminance. A Fast Fourier Transform (FFT) is computed of the resulting values, and the magnitudes of the resulting values are filtered with a CSF to an array of values. Finally the distance between the two images is computed by finding the Mean Square Error (MSE) of the values for each of the two images. This technique therefore measures similarity in Fourier amplitude between images.

Model 2 After Gervais et al: This model includes the effect of phase as well as magnitude in the frequency space representation of the image. Once again the luminances are normalised by dividing by the mean luminance. An FFT is computed producing an array of phases and magnitudes. These magnitudes are then filtered with an anisotropic CSF filter function constructed by fitting splines to psychophysical data.

Model 3 After Daly: In this model the effects of adaptation and non-linearity are combined in one transformation, which acts on each pixel individually. In the first two models each pixel has significant global effect in the normalisation by contributing to the image mean. Each luminance is transformed by an amplitude nonlinearity value. An FFT is applied to each transformed luminance and then they are filtered by a CSF (computed for a level of 50 cd/m2). The distance between the two images is then computed using MSE as in model 1.

The Visible Difference Predictor (VDP) is a perceptually based image quality metric proposed by Daly. Myskowski realised this metric had many potential applications in realistic image synthesis. He completed a comprehensive validation and calibration of VDP response via human psychophysical experiments. The VDP was tested to determine how close predictions come to subjective reports of visible differences between images by designing two human psychophysical experiments. Results from these experiments showed a good correspondence for shadow and lighting pattern masking and in comparison of the perceived quality of images generated as subsequent stages of indirect lighting solutions.

#### 8.2.2. Low-level perception-based error metrics

Perceptual error metrics have also been used in several other areas. Gibson and Hubbold[28] proposed a perception-driven hierarchical algorithm for radiosity used to decide when to stop hierarchy refinement. Links between patches are not refined anymore once the difference between successive levels of elements becomes unlikely to be detected perceptually. Gibson and Hubbold also applied a similar error metric to measure the perceptual impact of the energy transfer between two interacting patches, and to decide upon the number of shadow feelers that should be used in visibility test for these patches.

Perceptually-informed error metrics have also been successfully introduced to control the adaptive mesh subdivision and mesh simplification. Implementations have been done by Myszkowski et al.[50], Gibson and Hubbold[28], and Volevich et al.[76].

#### 8.2.3. Advanced Perception-Based Error Metrics

The scenario of embedding advanced HVS models into global illumination and rendering algorithms is very attractive, because computation can be perception-driven specifically for a given scene.

Bolin and Meyer[7] developed an efficient approximation of the Sarnoff Visual Discrimination Model (VDM), which made it possible to use this model to guide samples in a rendered image. Because samples were only taken in areas where there were visible artefacts, some savings in rendering time compared to the traditional uniform or adaptive sampling were reported.

Myszkowski[50] has shown some applications of the VDP to drive adaptive mesh subdivision taking into account visual masking of the mesh-reconstructed lighting function by textures.

Ramasubramanian et al.[61] have developed their own image quality metric which they applied to predict the sensitivity of the human observer to noise in the indirect lighting component. This made possible more efficient distribution of indirect lighting samples by reducing their number for pixels with higher spatial masking (in areas of images with high frequency texture patterns, geometric details, and direct lighting variations). All computations were performed within the framework of the costly path tracing algorithm, and a significant speedup of computations was reported compared to the sample distribution based on purely stochastic error measures.

A practical problem arises that the computational costs incurred by the HVS models introduce an overhead to the actual lighting computation, which may become the more significant the more rapid is the lighting computation. This means that the potential gains of such perception-driven computation can be easily cancelled by this overhead depending on many factors such as the scene complexity, per-

formance of a given lighting simulation algorithm for a given type of scene, image resolution and so on. The HVS models can be simplified to reduce the overhead, e.g., Ramasubramanian et al.[61] ignore spatial orientation channels in their visual masking model, but then underestimation of visible image artefacts becomes more likely. To prevent such problems and to compensate for ignored perceptual mechanisms, more conservative (sensitive) settings of the HVS models should be applied, which may also reduce gains in lighting computation driven by such models.

#### 8.2.4. Visible Differences Predictor

Although, substantial progress in physiology and psychophysics studies has been achieved in recent years, the Human Visual System (HVS) as the whole, and in particular, the higher order cognitive mechanisms, are not fully understood. Only the early stages of the visual pathway beginning with the retina and ending with the visual cortex are considered as mostly explored.

It is believed that the internal representation of an image by cells in the visual cortex is based on spatial frequency and orientation channels. The channel model explains such visual characteristics well as:

- The overall behavioural Contrast Sensitivity Function (CSF) - visual system sensitivity is a function of the spatial frequency and orientation content of the stimulus pattern.
- Spatial masking - detect ability of a particular pattern is reduced by the presence of a second pattern of similar frequency content.
- Sub-threshold summation - adding two patterns of sub-threshold contrast together can improve detect ability within a common channel.
- Contrast adaptation - sensitivity to selected spatial frequencies is temporarily lost after observing high contrast patterns of the same frequencies.
- The spatial frequencies after effects - as result of the eye adaptation to a certain grating pattern, other nearby spatial frequencies appear to be shifted.

Because of these favourable characteristics, the channel model provides the core of the most recent HVS models that attempt to describe spatial vision. The VDP is considered one of the leading computational models to predicting the differences between images that can be perceived by the human observer. The VDP receives as input a pair of images, and as output it generates a map of probability values, which characterize perceptibility of the differences.

The input target and mask images undergo an identical initial processing, Figure 22. At first, the original pixel intensities are compressed by the amplitude non-linearity based on the local luminance adaptation, simulating Weber's law-like behaviour. Then the resulting image is converted into the frequency domain and processing of CSF is performed. The resulting data is decomposed into the spatial frequency and orientation channels using the Cortex Transform, which is a pyramid-style, invertible, and computationally efficient image representation. Then the individual channels are transformed back to the spatial domain, in which visual masking is processed. For every channel and for every pixel, the elevation of detection threshold is calculated based on the mask contrast for that channel and that pixel. The resulting threshold elevation maps can be computed for the mask image, or mutual masking can be considered by taking the minimal threshold elevation value for the corresponding channels and pixels of the two input images. These threshold elevation maps are then used to normalize the contrast differences between target and mask images. The normalized differences are input to the psychometric function which estimates probability of detecting the differences for a given channel. This estimated probability value is summed across all channels for every pixel. Finally, the probability values are used to visualize visible differences between the target and mask images. It is assumed that the difference can be perceived for a given pixel when the probability value is greater than 0.75, which is standard threshold value for discrimination tasks. When a single numeric value is needed to characterize the differences between images, the percentage of pixels with probability greater than this threshold value is reported.

The main advantage of the VDP is a prediction of local differences between images (on the pixel level). The original Daly model also has some disadvantages, for example, it does not process chromatic channels in input images. However, in global illumination applications many important effects such as the solution convergence or the quality of shadow reconstruction can be relatively well captured by the achromatic mechanism, which is far more sensitive than its chromatic counterparts. The VDP seems to be one of the best existing choices for the prediction of image quality for various settings of global illumination solutions[2].

### 8.3. Comparing Real and Synthetic Images

#### 8.3.1. Human Visual Perception

A number of psychophysical experimental studies have demonstrated many features of how the HVS works. However, problems arise when trying to generalise these results for use in computer graphics. This is because, often, experiments are conducted under limited laboratory conditions and are typically designed to explore a single aspect of the visual system. Additionally, many previous classical psychophysical studies have contained factors either not practical, or just impossible, to recreate on a computer VDU. Instead of reusing information from previous psychophysical experiments, new experiments are needed which examine the HVS as a whole rather than trying to probe individual components.

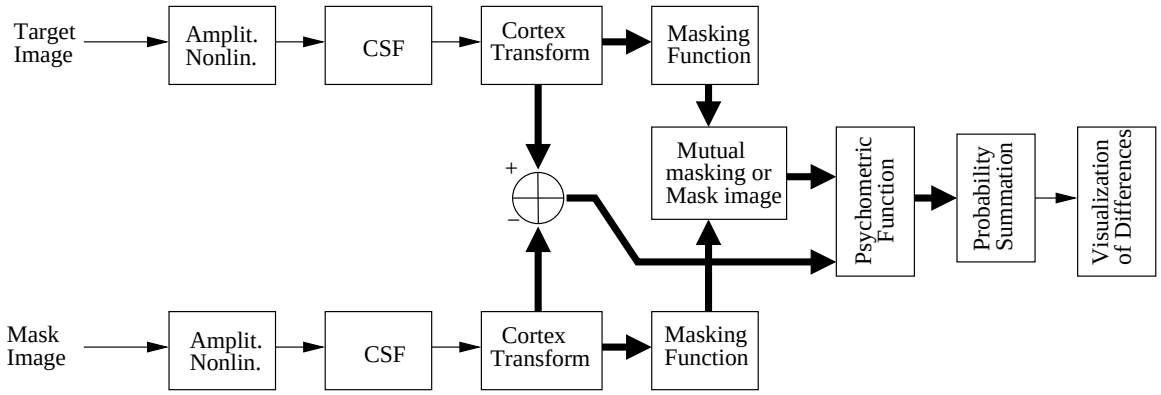



**Figure 22:** *Block diagram of the Visible Differences Predictor (heavy arrows indicate parallel processing of the spatial frequency and orientation channels.*

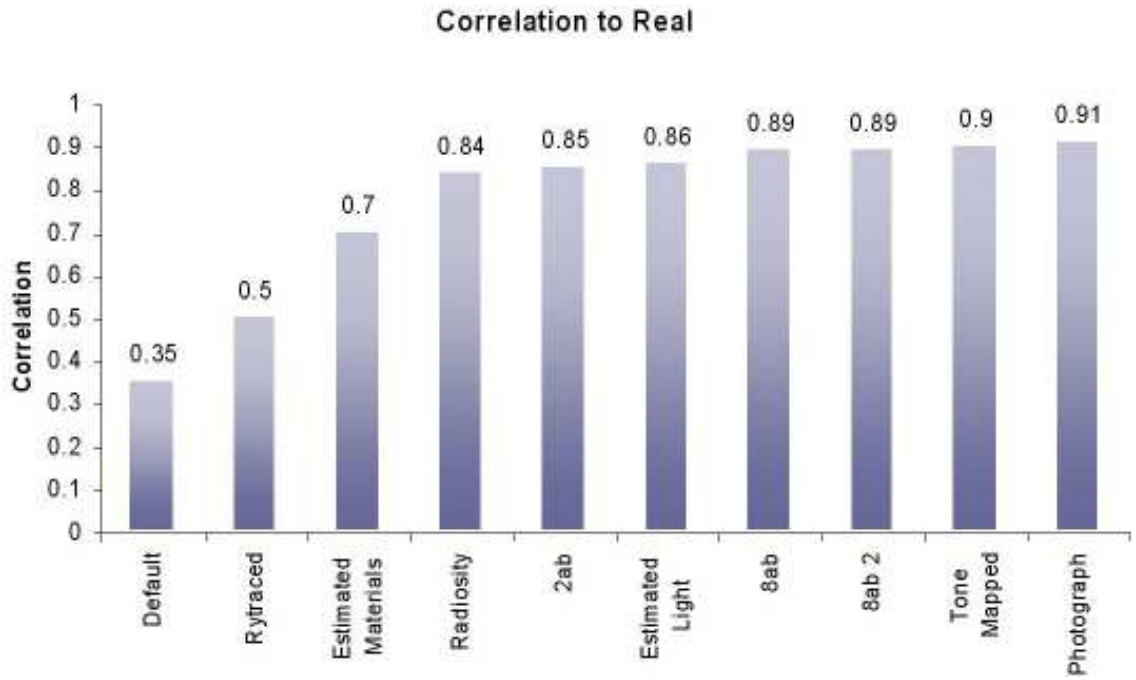



**Figure 24:** *Results obtained by McNamara et al. in lightness matching task experiments.*

### 8.3.2. Physical Comparisons

A number of experiments have been conducted at the University of Bristol where comparisons between made between real and synthetic images. These comparisons although comparing real and synthetic images have been task specific and have employed only simple controlled environments. McNamara et al.[47]. performed a series of experiments where subjects were asked to match lightness patches within the real world to those on a VDU, Figure 23. They discovered that a photograph of the real scene gave the highest perceptual match, with a high quality tone mapped rendered version coming a close second. A graph of their findings is shown in Figure 24. In all cases Radiance was used to render the images.

## 9. Tone Mapping and High Dynamic Range Imaging

The natural world presents our visual system with a wide range of colors and intensities. A starlit night has an average

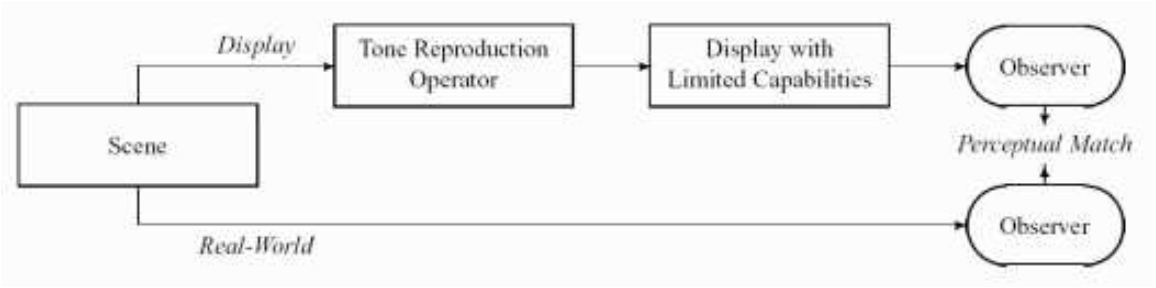



**Figure 25:** *Simple diagram of Tone Mapping.*

luminance level of around $10^{-3}$ $candelas/m^2$, and daylight scenes are close to $10^5$ $cd/m^2$.

Humans can see detail in regions that vary by 1:104 at any given adaptation level, over which the eye gets swamped by stray light (i.e., disability glare) and details are lost. Modern camera lenses, even with their clean-room construction and coated optics, cannot rival human vision when it comes to low flare and absence of multiple paths ("sun dogs") in harsh lighting environments. Even if they could, conventional negative film cannot capture much more range than this, and most digital image formats do not even come close. With the possible exception of cinema, there has been little push for achieving greater dynamic range in the image capture stage, because common displays and viewing environments limit the range of what can be presented to about two orders of magnitude between minimum and maximum luminance. A well-designed CRT monitor may do slightly better than this in a darkened room, but the maximum display luminance is only around 100 $cd/m^2$, which does not begin to approach daylight levels. A high-quality xenon film projector may get a few times brighter than this, but they are still two orders of magnitude away from the optimal light level for human acuity and color perception.

As a result of global illumination, images with huge dynamic ranges have become more common. Dealing with such values requires new file formats and more importantly devices able to display such range. The first requirement has been solved by the development of HDR file formats which allow the images to be stored in a more efficient way than storing three floating point numbers for each RGB. The RGBE file format[78] for example requires only 32 bit to store the whole luminance information.

Unfortunately it is still practicaly impossible to display these luminances on standard devices such as CRT monitors or printers. So how can the appearance of extremes of light and shadow be reproduced using only the tiny range of available display outputs?

Appearance-preserving transformations from scene to display, or tone reproduction operators, can solve this problem and were first described in the computer graphics literature by Tumblin and Rushmeier[75] as shown in Figure 25.

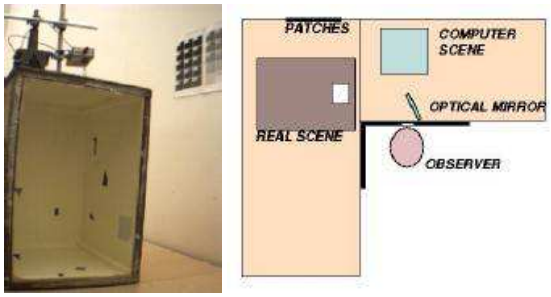



**Figure 23:** *Photograph and diagram of experimental setup employed by McNamara et al. in lightness matching task experiments.*

### 9.1. Tone Mapping

Tone-mapping algorithms rely on observer models that mathematically transform scene luminances into all the visual sensations experienced by a human observer viewing the scene, estimating the brain's own visual assessments. A tone reproduction operator tries to match the outputs of one observer model applied to the scene to the outputs of another observer model applied to the desired display image. Tumblin and Rushmeier[75] were the first to bring the issue of tone mapping to the computer graphics community. They offered a general framework for tone reproduction operators by concatenating a scene observer model with an inverse display observer model, and when properly constructed such operators should guarantee the displayed image is veridical: it causes the display to exactly recreate the visual appearance of the original scene, showing no more and no less visual content than would be discernible if actually present to see the original scene.

Unfortunately, visual appearance is still quite mysterious, especially for high contrast scenes, making precise and verifiable tone reproduction operators difficult to construct

and evaluate. Appearance, the ensemble of visual sensations evoked by a viewed image or scene, is not a simple one-to-one mapping from scene radiance to perceived radiance, but instead is the result of a complex combination of sensations and judgments, a set of well-formed mental estimates of scene illumination, reectance, shapes, objects and positions, material properties, and textures. Though all these quantities are directly measurable in the original scene, the mental estimates that make up visual appearance are not.

The most troublesome task of any basic tone reproduction operator is detail-preserving contrast. The human visual system (HVS) copes with large dynamic ranges through a process known as visual adaptation.

Local adaptation, the ensemble of local sensitivity-adjusting mechanisms in the human visual system, reveals visible details almost everywhere in a viewed scene, even when embedded in scenes of very high contrast. Although most sensations that humans perceive from scene contents, such as reectance, shape, color and movement can be directly evoked by the display outputs, large contrasts cannot. As shown in Figure 26, high contrasts must be drastically reduced for display, yet somehow must retain a high contrast appearance and at the same time keep visible in the displayed image all the low contrast details and textures revealed by local adaptation processes.

There are different reasons that make the tone mapping problem sometimes difficult to solve. The most obvious reason is that, as mentioned above the contrast ratio that can be produced by a standard CRT monitor is only about 100:1 which is much smaller that what can exist in the real world. Newspaper photographs achieve a maximum contrast of about 30:1, the best photographic prints can provide contrasts as high as 1000:1. In comparison, scenes that include visible light sources, deep shadows, and highlights can reach contrasts of 100000:1. Another reason that makes tone mapping operators fail in some cases is that the simplest ways to adjust scene intensities for display will usually reduce or destroy important details and textures.

### 9.1.1. Tone Mapping Operators

In the past decade quite a few authors have developed tone mapping operators to display HDR imagery. These algorithms can all be classified in two main categories: spatially uniform (non-local) and spatially varying (local). This is shown in Figure 27.

### 9.1.2. Local Operators

Humans are capable of viewing high contrasts scenes thanks to the local control sensitivity in the retina. This suggests that a position-dependent scale factor might reduce scene contrasts acceptably and allow them to be displayed on a low dynamic range device. This approach converts the original scene or real-world intensities to the displayed image intensities, using a position-dependent multiplying term.

Chiu et al.[11] addressed the problem of global visibility loss by scaling luminance values based on a spatial average of luminances in pixel neighborhoods. Very dark or bright areas are not clamped (like in the very first models) but are scaled according to their spatial location. Their approach provides excellent results on smoothly shaded portions of an image; however, any small bright feature in the image will cause strong attenuation of the neighboring pixels and surround the feature or high-contrast edge with a noticeable dark band or halo. This error occurs because the human eye is very sensitive to variation at high spatial frequencies.

Schlick[65] followed the work proposed by Chiu but this algorithm also reported problems with similar halo artifacts. Schlick used a first-degree rational polynomial function to map high-contrast scene luminances to display system values. This function works well when applied uniformly to each pixel of a high-contrast scene, and is especially good for scenes containing strong highlights. Next, he made an attempt to mimic local adaptation by locally varying a mapping function parameter; one method caused halo artifacts. Schlick concentrated mainly on efficiency and simplicity rather than improving the method mentioned above.

Rahman et al.[57] devised a full-color local scaling and contrast reduction method using a multiscale version of Land's "retinex" theory of color vision. Retinex theory estimates scene reflectances from the ratios of scene intensities to their local intensity averages. Jobson, Rahman, and colleagues also use Gaussian low-pass filtering to find local multiplying factors, making their method susceptible to halo artifacts. They divide each point in the image by its low-pass filtered value, then take the logarithm of the result to form a reduced contrast "single-scale retinex." To further reduce halo artifacts they construct a "multiscale retinex" from a weighted sum of three single-scale retinexes, each computed with different sized filter kernels, then apply scaling and offset constants to produce the display image. These and other constants give excellent results for the wide variety of 24bit RGB images used to test their method, but it is unclear whether these robust results will extend to floating-point images whose maximum contrasts can greatly exceed 255:1.

Pattanaik et al.[53] proposed a tone reproduction algorithm that takes into account representations of pattern, luminance and color processing in the Human Visual System. The model accounts for changes of perception at threshold and suprathresholds levels of brightness. This tone mapping algorithm also allows chromatic adaptation as well as luminance adaptation, Figure **??**. It however doesn't include any time adaptation models.

Recently Reinhard? proposed an operator that is based on photographic practice using a system called the zone system which divides the scenes luminances into 11 printing zones. The zones go from black (zone 0) to white (zone 10). Then a luminance reading for a middle gray is taken and is assigned to zone 5. The dynamic range is captured by reading

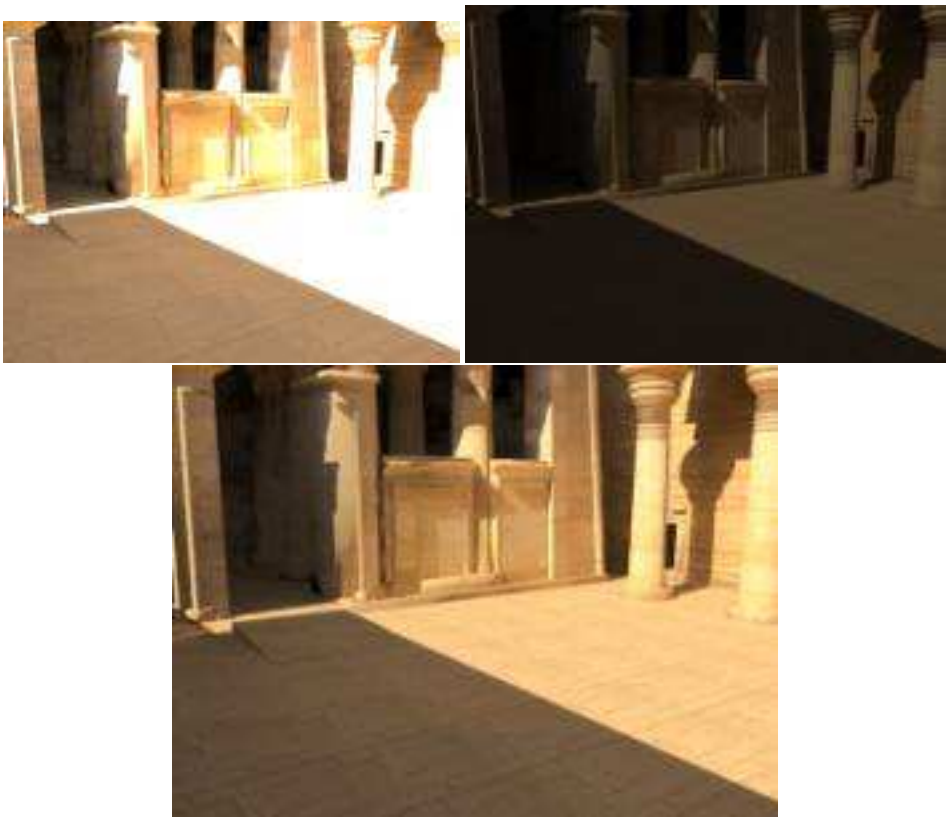


**Figure 26:** *Top images show the dynamic range, bottom is the tone mapped image.*

light and dark regions. This operator firstly applies a scaling to the entire image to reduce the dynamic range and then modifies locally the contrast of some regions by highlighting or darkening to improve the overall visibility. There is also a global version of the operator which tries to simulate the "dodging and burning" techniques used in photography.

### 9.1.3. Global Operators

Most imaging systems do not imitate local adaptation. Instead, almost all image synthesis, recording, and display processes use an implicit normalizing step to map the original scene intensities to the target display intensities without disturbing any scene contrasts that fall within the range of the display device. This normalizing consists of a single constant multiplier. Image normalizing has two important properties: it preserves all reproducible scene contrasts and it discards the intensities of the original scene or image.

Contrast, the ratio of any two intensities, is not changed if both intensities are scaled by the same multiplier. Normalizing implicitly assumes that scaling does not change the appearance, as if all the perceptually important information were carried by the contrasts alone, but scaling display intensities can strongly affect a viewer's estimates of scene contrasts and intensities. Although this scaling is not harmful for many well-lit images or scenes, discarding the original intensities can make two scenes with different illumination levels appear identical. Normalizing also fails to capture dramatic appearance changes at the extremes of lighting, such as gradual loss of color vision, changes in acuity, and changes in contrast sensitivity.

Tumblin and Rushmeier[75] tried to capture some of these light dependent changes in appearance by describing a "tone reproduction operator," which was built from models of human vision, to convert scene intensities to display intensities. They offered an example operator based on the suprathreshold brightness measurements made by Stevens and Stevens[72] who claimed that an elegant power-law relation exists between luminance, adaptation luminance, and perceived brightness. Tumblin and Rushmeier's used the results of Stevens and Stevens and tried to preserve brightness in a scene. However it had some large limitations: images or scenes that approach total darkness processed with their method are displayed as anomalous middle gray images instead of black, and display contrasts for very bright images ($> 100\ cd/m^2$) are unrealistically exaggerated.

Soon afterwards Ward[77] presented a much simpler ap-

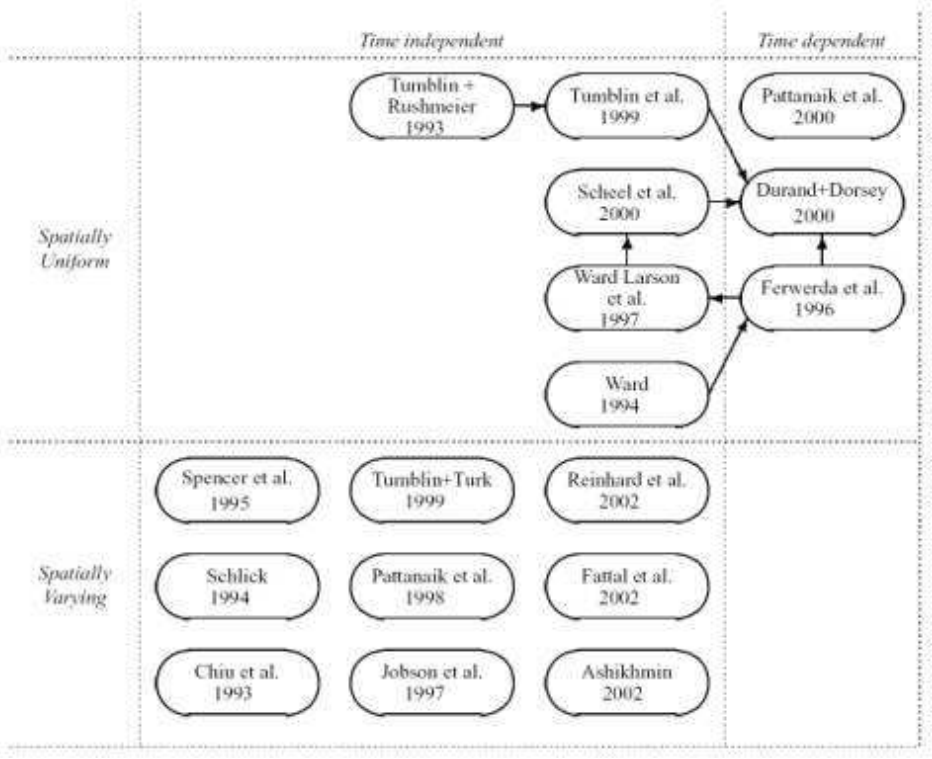



**Figure 27:** *Taxonomy of tone mapping operators (after Devlin et al.[20]).*

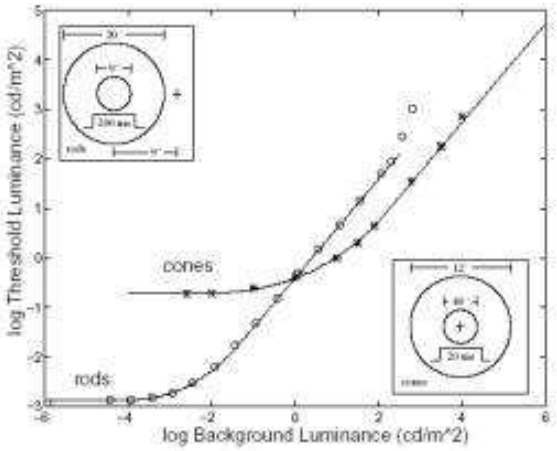



**Figure 28:** *Detection thresholds over the full range of vision.*

proach to appearance modeling that also provided a better way to make dark scenes appear dark and bright scenes appear bright on the display. The idea behind this operator is that visibility is preserved which insures that the smallest perceptible difference in a real scene corresponds to the smallest perceptible difference in the image.

Ferwerda et al.[23] offered an extended appearance model for adaptation that successfully captured several of its most important visual effects. This operator takes into account the transition for achromatic night vision and chromatic day vision. This is achieved by modeling the gradual transition from cone-mediated daylight vision to rod-mediated night vision. This method accounts for change in color sensitivity acuity as a function of intensity in the scene. Like Ward, they converted original scene or image intensities to display intensities with a multiplicative scale factor, but they determined their multiplier values from a smooth blending of increment threshold data for both rods and cones in the retina, as shown in Figure 28. This method also provides a simple method to mimic the time course of adaptation for both dark-to-light and light-to-dark transitions.

More recently Ward et al.[78] published a new and impressively comprehensive tone reproduction operator based on iterative histogram adjustment and spatial filtering processes. Their operator reduces high scene contrasts to match display abilities, and also ensures that contrasts that exceed human visibility thresholds in the scene will remain visible on the display (bottom image in Figure 26). They model some foveally dominated local adaptation effects, yet completely avoid halo artifacts or other forms of local gradient reversals, and include new locally adapted models of glare,

color sensitivity, and acuity similar to those used by Ferwerda et al.[23].

In 1999 Tumblin et al.[74] proposed two methods to display high contrast images on low dynamic range displays by imitating some of the human visual systems' properties. One method, based on HVS layer models, creates images in lighting layers and surface properties. The algorithm aims to preserve scene visibility. This is achieved by scaling all the luminance levels and compressing them while preserving the reflectance and transparency layers. The main limitation with this process is that it only works with rendered images where all the layer information can be retrieved during the rendering process. Tumblin's second method, know as the foveal method, interactively adjusts the detail visibility in the fovea area whilst compressing other parts of the image.

The user can use the mouse to click on any area of an image and the algorithm tone maps the surrounding area base on the local luminance levels.

A recent operator was proposed by Pattanaik et al.[54]. This time dependent algorithm allows both static or dynamic images (photographs or rendered) to be tone mapped and is based on a perceptual model proposed by Tumblin and Rushmeier. It also includes an eye adaptation model to represent lightness and color. This operator is original since it accepts a variety of scenes and luminance levels and it takes into account various adaptation factors. All the human eye properties are obtained from widely accepted color science and psychology literature making this operator ideal for dynamic scenes. Using Hunt's[42] colour model for static vision they include time dependent effects such as neural response and color bleaching effects. They main limitation of this operator however is that it does not include a local eye-adaptation approach which is very important to faithfully represent visual appearance.

A few other computer graphics researchers have modeled the appearance of extremely bright, high-contrast scene features by adding halos, streaks, and blooming effects to create the appearance of intensities well beyond the abilities of the display. Nakamae et al. proposed that the star-like streaks seen around bright lights at night are partly due to diffraction by eyelashes and pupils, and they presented a method to calculate these streaks in RGB units, implicitly normalizing them for display. Later Spencer et al.[71] presented an extensive summary of the optical causes and visual effects of glare and modeled their appearance by using several adjustable low-pass filters on the intensities of the original scene. Small, extremely bright light sources that cover only a few pixels, such as street lights at night or the sun leaking through a thicket of trees, are expanded into large, faintly colored, glare-like image features that have a convincing and realistic appearance. Despite progress in modelling the light-dependent changes in appearance that occur over the entire range of human vision, few methods offer the substantial contrast reduction needed to display these images without truncation or halo artefacts.

#### 9.1.4. Perceptual Vs. Non-Perceptual

Some algorithms use perceptual data to simulate reality, others simply attempt to compress the range purely by a mathematical approach with the aim of obtaining the maximum visibility on the display device. This can be useful if the TMO operator is simply used for visualization purposes in which case displaying all the possible values can be satisfactory. However, in all those cases where Tone Mapping tries to simulate reality, the implementation of the algorithm should be based on perceptual data. A few of the operators published try to simulate human visibility by mathematically modeling some property of the HVS such as eye-adaptation, color visibility at different photopic or scotopic light level, visual acuity.

One of the main limitations of tone mapping is that the displayed result is static. Although some algorithms take into account human visibility factors, only very few operators allow to dynamically modify the image based on human eye models. It is important to decide what are the purposes of a particular algorithm. If for example it is important to visualize all the luminance levels in a scene "in one go" then most operators satisfy this. However, if an eye-simulation is required then a dynamic model based on adaptation may be more accurate.

## 10. Conducting Psychophysical Experiments

Psychophysical experiments can be used in attempt to answer such questions as: How realistic is this synthesised image?' In order to investigate answers to subjective questions such as this, data needs to be collected. Typically this data is in some numerical form usually derived from a questionnaire completed by the participant or the researcher during the experiment

A mass of data can be summarised or different sets of data can be compared by the calculation of appropriate statistics which will provide answers to vital questions such as the one proposed above. Thus, statistical analysis is the most useful technique for helping the researcher find answers to the questions set.

But how can 'realism' be measured and who can guarantee that the observed measures can easily be translated into norms of human perception of reality? This is certainly a tricky question.

### 10.1. Design

For many, it is the beauty of ideas and hypothesis testing that keeps psychophysical experiments being conducted. Suffice to say that a flawed design will derail even the most impressive theory and hypothesis, whereas appropriate, well-thought-out designs usually lead to informative research and

compelling findings. Thus, the design of any psychophysical experiment may be its most important part, the one in which the whole study and outcome will be based. Nearly everything else in the actual experiment depends on it. Moreover, the design dictates many features of method and data analysis. Perhaps most important, design more than any other quality, with the exception of the data itself, determines what conclusions can and cannot be drawn. It is apparent, in short, that developing a good experimental design and describing it clearly and informatively is an essential step in writing and presenting an interesting research finding.

There is little doubt that psychophysical experiments are a lot more complex nowadays than they once were. Whereas in the not too distant past a few basic designs sufficed for most questions, the accumulation of a sizable literature and growing technical complexity of the field has dictated that contemporary researchers develop and become familiar with diverse designs[62]. Thus, whereas it once might have been possible to fully describe and explain a standard research design with one or two phrases, a bit more attention is now needed. Any basic research methods textbook explains the advantages and disadvantages of most of these designs. What is not as readily apparent, given their complexity and diversity, is how to convey the essential features of a given design clearly yet efficiently. Although there are similarities, each type of design necessitates its own specifications. Consequently, the information contained in the ideal description varies from one design to another. For instance, experimental designs can be between-participants, within-participants or mixed.

In a between-participants design, all participants take part in one and only one cell of the design, whereas in the latter the same participants engage in multiple conditions. Between-participants designs require mention of how participants were assigned to conditions, randomly or by some other procedure. In these designs, experimental conditions are specified according to the independent variables (IV). Each independent variable has two or more conditions or levels. If there is more than one independent variable, the design is called a factorial design. It is common to refer to factorial designs by the number of levels of each independent variable or factor. For example, a 3 x 2 x 2 factorial design has three independent variables, one with three levels and two with two levels each, resulting in 12 combinations (or cells). Design statements should always be clear about the independent variables, the levels of each independent variable, and the factorial structure that organised them, which may not be apparent.

Experiments with a single dependent variable (DV) are called univariate, whereas those with multiple dependent variables are called multivariate. In multivariate designs, it is generally useful to describe how the dependent variables are organised, for example, whether they assess separate constructs or are essentially parallel. As we have already mentioned, there are experimental designs in which the same person participates in more than one condition, the so called within-participants design. For example, each participant can be engaged in a preference task for a chair under several different conditions (photograph of a chair, rendered image of a chair etc). In the experimental design, because each individual engaged in multiple conditions, the order of administration is certainly important. Common strategies for contending with order effects include counterbalancing (an equal number of participants experience each condition in each serial position), partial randomisation (in which only certain orderings chosen to control for the most plausible effects, are used), and randomisation, as well as leaving order fixed.

Having made these various points, there is little doubt that the design section is the most critical part of any psychophysical experiment, leaving little room for error or omission. The importance of elegant, creative, and timely theorising notwithstanding, behavioural science at its core is all about evidence, and how well it supports a given set of ideas and hypotheses. Such support is a direct consequence of research design. Good designs provide a strong foundation for the validity of conclusions by fostering particular explanations and ruling out others. Poor designs are either inappropriate to the conclusions or invite conceptual ambiguity. In short, the extent to which a study adds to knowledge depends as much on design as anything else.

The first question any reader and reviewer should ask is whether the obtained results of a study validly and unambiguously lead to the conceptual conclusions that a researcher advocates. If the answer is no, or even maybe not, readers are likely to raise substantial questions about the research's contribution to the current domain of human perception, irrespective of its theoretical polish and numerous highly significant results which support the research hypothesis. Design is a big part of that judgement, although certainly not the only one, and it is therefore generally a good idea to prepare a design section with sceptical readers in mind.

### 10.2. Planning

Suffice to say that any psychophysical experiment needs to be planned carefully. But what do we mean by 'planning our research'? There are some basic steps that need to be followed in order to be sure that the experiment has been planned successfully and that the outcome is inevitably going to be valid and applicable. The decision areas facing anyone about to conduct some research are:

1. What will be measured and how, exactly?
2. Who will be studied?
3. How will the data gathered be used to demonstrate a real difference?

Decision 1 concerns the precise measurement of variables. For instance we need to give a specific means by which to measure 'realism'. Variables are things which vary and need to be precisely defined in the research project.

Decision 2 concerns the participants that we are going to test. For instance what is the advantage of using the same group of people for each condition?

Decision 3 is probably the hardest. How do we know when a discovered difference is a real one and not just the result of random variation? For instance, when do we become convinced that people do not perceive differences between real and synthetic images? With reference to the goal of the perceiving realism, there is little doubt that a number of psychophysical experiments need to be conducted in order to be able to validate and examine the realism of synthetic images and people's perception of them. The outcome of carefully planned and organized psychophysical experiments will lead to an important added-value in image synthesis, by enhancing the realism of augmented environments through consistent illumination of a scene containing real and virtual objects. Currently questionnaires are the best tool available for data collection in order to obtain participant's responses to the questions set. For the purposes of the example being considered here, participants can be tested repeatedly (use a within-subjects design) in order to compare their responses to the various stimuli presented to them i.e. a rendered chair, a photograph of a real chair etc. In that way, we will have a measure of their perception and also some data in a form which can easily be presented in a numerical form and analysed in a statistical package, such as SPSS.

### 10.3. Questionaires

Although designing a questionnaire might sound easy, questionnaires do not emerge fully-fledged and thus are quite difficult to compose. Questionnaires have to be composed and tried out, improved and then tried out again, often several times over, until we are certain that they can do the job for which they are needed. This whole lengthy process of designing and trying out questions and procedures is usually referred to as a 'pilot study'. Piloting can help us not only with the wording of questions but also with procedural matters such as the design of a letter of introduction, the ordering of question sequences and the reduction of non-response rates. We should realize from the beginning that pilot studies are time-consuming, but avoiding or skimping on this is likely to lead to errors in the final experiments. Although there are many different methods of data collection such as mail questionnaires and group administered questionnaire to name but a few, for our purposes, self-administer questionnaires seems to be the most promising method since they ensure a high response rate, accurate sampling and a minimum of interview bias, while permitting interviewer assessments, providing necessary explanations (but not the interpretation of questions) and giving the benefit of a degree of personal contact. Another important element of questionnaires is that of 'question type'. Open ended and closed questions have both a number of advantages and disadvantages all of which must be considered by the researcher and be adjusted by the needs of the research purpose. For instance open-ended questions are time-consuming whereas closed questions require little time on behalf of both the participant and the experimenter (analysis). It is imperative to mention at this point that questionnaires are not the only tools available for collecting data. For instance, we can use reaction times in a recognition test in order to measure differences between or within participants. Taken together, there are a numerous methods by which a researcher can gather data and provide valid results for the question set. However, one must keep in mind that psychophysical experiments need preparation and critical thinking in order to be conducted appropriately and provide adequate results.

### 10.4. Example

Figures 29 to 31 show an example questionnaire, which should give some ideas as to how such a form should be laid out. This questionnaire was used in a study of inattentional blindness[46]. The specific question being consider was: While performing a visual task, would the participants notice any changes any changes in their environment if something was changed during the course of the experiment?

The participants we informed, as stated on the questionnaire, that their task was to search for an object in a picture. They were told that the difference between the experiments was the type of music that was played. In addition to the participant, there was the experimenter and an assistant in the room at the time.

During the course of the experiment, the lights were "accidentally" turned off and the assistant was changed. The real purpose of the experiment was to determine if any of the participants noticed the change to the assistant.

Full results of this experiment have yet to be published, but 100% of the participants failed to notice when two males were used as the assistants and, perhaps surprisingly, 85% of the participants failed to notice when the assistant was changed from a male to a female or vice versa.

## 11. Summary

In this tutorial, we have described a system that allows us to generate visually realistic Augmented images at interactive rates. The tutorial has covered the techniques we use for data capture, object shading and shadow generation, and has also discussed some of the important issues that must be considered when trying to assess the perceptual fidelity of synthetic images.

We have shown that we can generate subjectively realistic

augmented images at interactive rates for a variety of different real-world lighting environments including both interior and natural illumination. Our system is also capable of trading image accuracy against frame-rate by approximating shadows using different numbers of shadow-maps. As the number of shadow blending passes (and hence frame generation time) increases, the result rapidly approaches the quality obtained using traditional non-real-time approaches.

There are currently limitations in our system on the types of light sources that can be modelled. For example, we are unable to render shadows cast by direct sunlight, or other types of directional illumination. There is nothing inherent in the rendering algorithm preventing this, but our current methods of data capture (Section 4) are not able to distinguish between directional and diffuse sources of light in the scene. We also assume that all surfaces onto which shadows are cast are diffuse, although is not a fundamental limitation of the algorithm. Because we pre-compute the radiance reduction caused by the occlusion of each source of light (Section 6.1), a view-dependent evaluation of this could account for non-diffuse reflectance properties. However, such extensions are left as future work, mainly because of the complexity of recovering non-diffuse surface reflectance data for real-world environments[81, 6].

Although we have presented examples showing augmentation of static images, our shadow generation algorithm is not view-dependent in any way, and the techniques presented here could also be applied to moving cameras [67]. Finally, the overall rendering quality will be enhanced by the appearance of floating-point graphics pipelines in the next generation of computer graphics hardware. This will reduce rounding errors that can sometimes occur when blending large numbers of very faint shadows into the background image.

**Acknowledgements**

We would like to thank our colleagues Maria Karipoglou, Patrick Ledda and Peter Longhurst at the University of Bristol, and Jon Cook, Toby Howard and Roger Hubbold at the University of Manchester for assistance with this work and the preparation of these notes. We are also grateful to the other ARIS project partners for their support and assistance (Fraunhofer IGD, Intracom, INRIA-Loria, and Athens Technology Center).

We would like to acknowledge the European Union for funding this work, as part of the ARIS project (IST-2000-28707). The bunny model used in Figure 16 is available from the Stanford 3D Scanning Repository.

**References**

1. http://aris-ist.intranet.gr/. ARIS project, IST-2000-28707, 2001-2004. 2
2. http://www.u-aizu.ac.jp/labs/csel/vdp/. Web page with documentation of the VDP validation experiments. 25
3. 3rdTech. Deltasphere 3d scene digitizer. http://www.3rdtech.com. 4
4. P. A. Beardsley, P. H. S. Torr, and A. Zisserman. 3D model aquisition from extended image sequences. In *Proc. 4th European Conference on Computer Vision, LNCS 1065, Cambridge*, pages 683–695, 1996. 4
5. S. Becker and V. M. Bove Jr. Semi-automatic 3-d model extraction from uncalibrated 2-d camera views. *SPIE Symposium on Electronic Imaging: Science & Technology, San Jose*, February 1995. 4
6. S. Boivin and A. Gagalowicz. Image-based rendering of diffuse, specular and glossy surfaces from a single image. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pages 107–116, August 2001. 9, 34
7. M. R. Bolin and G. W. Meyer. A perceptually based adaptive sampling algorithm. In *Proceedings of SIGGRAPH 98*, Computer Graphics Proceedings, Annual Conference Series, pages 299–310, July 1998. 24
8. S. Bougnoux and L. Robert. Totalcalib: A fast and reliable system for off-line calibration of image sequences. In *Proc. IEEE International Conference on Computer Vision and Pattern Recognition*, 1997. 4
9. S. Brabec and H.-P. Seidel. Single sample soft shadows using depth maps. In *Proc. Graphics Interface*, pages 219–228, May 2002. 5
10. L. S. Brotman and N. I. Badler. Generating soft shadows with a depth buffer algorithm. *IEEE Computer Graphics & Applications*, 4(10):71–81, October 1984. 5
11. K. Chiu, M. Herf, P. S. Shirley, S. Swamy, C. Wang, and K. Zimmerman. Spatially nonuniform scaling functions for high contrast images. In *Graphics Interface '93*, pages 245–253, May 1993. 28
12. R. Cipolla and E. G. Boyer. 3d model acquisition from uncalibrated images. *Proc. IAPR Workshop on Machine Vision Applications, Chiba, Japan*, pages 559–568, November 1998. 4
13. R. Cipolla, T. Drummond, and D.P. Robertson. Camera calibration from vanishing points in images of architectural scenes. *Proc. British Machine Vision Conference, Nottingham*, 2:382–391, September 1999. 5
14. R. Cipolla, D. P. Robertson, and E. G. Boyer. Photobuilder – 3d models of architectural scenes from uncalibrated images. In *Proc. IEEE International Conference on Multimedia Computing and Systems*, volume 1, pages 25–31, 1999. 4
15. M. F. Cohen and J. R. Wallace. *Radiosity and Realistic Image Synthesis*. Academic Press Professional, Boston, MA, 1993. 4, 5, 9, 12

16. A. Criminisi, I. Reid, and A. Zisserman. Single view metrology. *International Journal of Computer Vision*, 40(2):123–148, November 2000. 4

17. P. Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of SIGGRAPH 98*, Computer Graphics Proceedings, Annual Conference Series, pages 189–198, Orlando, Florida, July 1998. 2, 5, 8, 11, 13, 19

18. P. E. Debevec and J. Malik. Recovering high dynamic range radiance maps from photographs. In *Proceedings of SIGGRAPH 97*, Computer Graphics Proceedings, Annual Conference Series, pages 369–378, Los Angeles, California, August 1997. 2, 5, 8

19. P. E. Debevec, C. J. Taylor, and J. Malik. Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. In *Proceedings of SIGGRAPH 96*, Computer Graphics Proceedings, Annual Conference Series, pages 11–20, New Orleans, Louisiana, August 1996. 4

20. K. Devlin, A. Chalmers, A. Wilkie, and W. Purgathofer. Tone reproduction and physically based spectral rendering. In *State of the Art Report, Eurographics 2002*, pages 101–123, Saarbrucken, Germany, September 2002. 30

21. G. Drettakis, L. Robert, and S. Bougnoux. Interactive common illumination for computer augmented reality. In *Proc. Eurographics Rendering Workshop 1997*, pages 45–56, St. Etienne, France, June 1997. 5

22. G. Drettakis and F. X. Sillion. Interactive update of global illumination using a line-space hierarchy. In *Proceedings of SIGGRAPH 97*, Computer Graphics Proceedings, Annual Conference Series, pages 57–64, Los Angeles, California, August 1997. 5, 12

23. J. A. Ferwerda, S. N. Pattanaik, P. S. Shirley, and D. P. Greenberg. A model of visual adaptation for realistic image synthesis. In *Proceedings of SIGGRAPH 96*, Computer Graphics Proceedings, Annual Conference Series, pages 249–258, August 1996. 22, 30, 31

24. A. W. Fitzgibbon and A. Zisserman. Automatic camera recovery for closed or open image sequences. In *Proc. European Conference on Computer Vision*, pages 311–326. Springer-Verlag, June 1998. 4

25. A. Fournier, A. S. Gunawan, and C. Romanzin. Common illumination between real and computer generated scenes. In *Graphics Interface '93*, pages 254–262, May 1993. 5

26. M. Garland and P. S. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of SIGGRAPH 97*, Computer Graphics Proceedings, Annual Conference Series, pages 209–216, August 1997. 16

27. S. Gibson, J. Cook, T. L. J. Howard, and R. J. Hubbold. Rapid shadow generation in real-world lighting environments. In *Rendering Techniques 2003 (Proceedings of the Eurographics Rendering Symposium)*, June 2003. 1

28. S. Gibson and R. J. Hubbold. Perceptually-driven radiosity. *Computer Graphics Forum*, 16(2):129–140, June 1997. 24

29. S. Gibson, Roger J. Hubbold, J. Cook, and T. L. J. Howard. Interactive reconstruction of virtual environments from video sequences. *Computers & Graphics*, 27(3), April 2003. 6

30. S. Gibson and A. Murta. Interactive rendering with real world illumination. In *Rendering Techniques 2000: 11th Eurographics Workshop on Rendering*, pages 365–376, June 2000. 5

31. A. Gilchrist. *Brightness and Transparency*. Hillsdale: Lawerence Erlibaum Associates, 1996. 23

32. A. S. Glassner. *Principles of Digital Image Synthesis*. Morgan Kaufmann, San Francisco, CA, 1995. 4

33. X. Granier and G. Drettakis. Incremental updates for rapid glossy global illumination. *Computer Graphics Forum*, 20(3):268–277, 2001. 5

34. G. Greger, P. S. Shirley, P. M. Hubbard, and D. P. Greenberg. The irradiance volume. *IEEE Computer Graphics & Applications*, 18(2):32–43, March-April 1998. 5, 9

35. E. A. Haines. A shaft culling tool. *Journal of Graphics Tools*, 5(1):23–26, 2000. 12

36. E. A. Haines and T. Möller. Real-time shadows. In *Game Developers Conference*, March 2001. 4

37. S. El. Hakim. A practical approach to creating precise and detailed 3d models from single and multiple views. In *International Archives of Photogrammetry and Remote Sensing, B5A, Commission V*, volume 33, pages 122–129, July 2000. 4

38. R. I. Hartley. Euclidean reconstruction from uncalibrated views. *Applications of Invariance in Computer Vision, LNCS-Series*, 825:237–256, 1994. 4

39. R. I. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge, UK, 2000. 5

40. S. Hecht. The visual discrimination of intensity and the weber-fechner law. *Journal of General Physiology*, 7, 1924. 12

41. P. S. Heckbert and M. Herf. Simulating soft shadows

with graphics hardware. Technical Report CMU-CS-97-104, CS Department, Carnegie Mellon University, January 1997. 11

42. G. Hunt. *The Reproduction of Colour*. Kings Langley: Fountain Press, 3rd edition, 1975. 5th edition is now available. 31

43. A. Keller. Instant radiosity. In *Computer Graphics (ACM SIGGRAPH '97 Proceedings)*, volume 31:3, pages 49–56, 1997. 5

44. L. Latta and A. Kolb. Homomorphic factorization of brdf-based lighting computation. *ACM Transactions on Graphics*, 21(3):509–516, July 2002. 1, 5, 11

45. C. Loscos, M.-C. Frasson, G. Drettakis, B. Walter, X. Granier, and P. Poulin. Interactive virtual relighting and remodeling of real scenes. In *Proc. Eurographics Rendering Workshop 1999*, Granada, Spain, June 1999. 5

46. A. Mack and I. Rock. *Inattentional Blindness*. Massachusetts Institute of Technology Press, 1998. 33, 39

47. A. McNamara. *Comparing Real and Synthetic Scenes using Human Judgements of Lightness*. PhD thesis, University of Bristol, 2000. 23, 26

48. G. W. Meyer, H. E. Rushmeier, M. F. Cohen, D. P. Greenberg, and K. E. Torrance. An experimental evaluation of computer graphics imagery. *ACM Transactions on Graphics*, 5(1):30–50, January 1986. 23

49. T. Mitsunaga and S. K. Nayar. Radiometric self calibration. In *IEEE Conf on Computer Vision and Pattern Recognition*, 1999. 8

50. K. Myszkowski. The visible differences predictor: Applications to global illumination problems. In *Eurographics Rendering Workshop 1998*, pages 223–236, June 1998. 24

51. R. Ng, R. Ramamoorthi, and P. Hanrahan. All-frequency shadows using non-linear wavelet lighting approximation. In *Proceedings of SIGGRAPH 2003*, Computer Graphics Proceedings, Annual Conference Series, San Diego, California, July 2003. 1

52. S. Parker, P. S. Shirley, and B. Smits. Single sample soft shadows. Technical Report UUCS-98-019, Computer Science Department, University of Utah, October 1998. 5

53. S. N. Pattanaik, J. A. Ferwerda, M. D. Fairchild, and D. P. Greenberg. A multiscale model of adaptation and spatial vision for realistic image display. In *Proceedings of SIGGRAPH 98*, Computer Graphics Proceedings, Annual Conference Series, pages 287–298, July 1998. 28

54. S. N. Pattanaik, J. E. Tumblin, H. Yee, and D. P. Greenberg. Time-dependent visual adaptation for realistic image display. In *Proceedings of ACM SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, pages 47–54, July 2000. 31

55. M. Pollefeys, R. Koch, and L. van Gool. Structure and motion from image sequences. In Grun Kahmen, editor, *Proc. Conference on Optical 3-D Measurement Techniques V, Vienna, Austria*, pages 251–258, October 2001. 4

56. P. Poulin, M. Ouimet, and M. C. Frasson. Interactive modeling with photogrammetry. In *Proc. Eurographics Workshop on Rendering, Vienna, Austria*, July 1998. 4

57. Z. Rahman, D. J. Jobson, and G. A. Woodell. Multiscale retinex for color image enhancement. In *Proceedings, International Conference on Image Processing*, volume 3, pages 1003–1006, Lausanne, Switzerland, September 1996. 28

58. R. Ramamoorthi and P. Hanrahan. An efficient representation for irradiance environment maps. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pages 497–500, August 2001. 9

59. R. Ramamoorthi and P. Hanrahan. On the relationship between radiance and irradiance: Determining the illumination from images of a convex lambertian object. In *Journal of the Optical Society of America*, volume 18:10, pages 2448–2459, October 2001. 9

60. R. Ramamoorthi and P. Hanrahan. Frequency space environment map rendering. *ACM Transactions on Graphics*, 21(3):517–526, July 2002. 1, 11

61. M. Ramasubramanian, S. N. Pattanaik, and D. P. Greenberg. A perceptually based physical error metric for realistic image synthesis. In *Proceedings of SIGGRAPH 99*, Computer Graphics Proceedings, Annual Conference Series, pages 73–82, August 1999. 24, 25

62. H. T. Reis and K. Stiller. Publication trends in jpsp: A three decade review. *Personality and Social Psychology Bulletin*, 15:465–472, 1992. 32

63. H. Rushmeier, G. W. Larson, C. Piatko, P. Sanders, and B. Rust. Comparing real and synthetic images: Some ideas about metrics. In *Eurographics Rendering Workshop 1995*, pages 82–91, June 1995. 24

64. I. Sato, Y. Sato, and K. Ikeuchi. Acquiring a radiance distribution to superimpose virtual objects onto a real scene. *IEEE Transactions on Visualization and Computer Graphics*, 5(1):1–12, January - March 1999. 5

65. C. Schlick. Quantization techniques for high dynamic range pictures. *Photoreaslistic Rendering Techniques*, pages 7–20, 1990. 28

66. F. X. Sillion and C. Puech. *Radiosity and Global Illumi-*

*nation*. Morgan Kaufmann, San Francisco, CA, 1994. 4, 5, 12

67. G. Simon and M.-O. Berger. Reconstructing while registering: a novel approach for markerless augmented reality. In *International Symposium on Mixed and Augmented Reality - ISMAR'02, Darmstadt, Germany*, 2002. 34

68. P.-P. Sloan, J. Hall, J. Hart, and J. Snyder. Clustered principal components for precomputed radiance transfer. *ACM Transactions on Graphics*, 22(3), July 2003. 5, 11

69. P.-P. Sloan, J. Kautz, and J. Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. *ACM Transactions on Graphics*, 21(3):527–536, July 2002. 1, 5, 11

70. C. Soler and F. X. Sillion. Fast calculation of soft shadow textures using convolution. In *Proceedings of SIGGRAPH 98*, Computer Graphics Proceedings, Annual Conference Series, pages 321–332, July 1998. 5

71. G. Spencer, P. S. Shirley, K. Zimmerman, and D. P. Greenberg. Physically-based glare effects for digital images. In *Proceedings of SIGGRAPH 95*, Computer Graphics Proceedings, Annual Conference Series, pages 325–334, August 1995. 31

72. S. Stevens and J. C. Stevens. Brightness function: Parametric effects of adaptation and contrast. *Journal of the Optical Society of America*, 50(11), November 1960. 29

73. P. Tole, F. Pellacini, B. Walter, and D. P. Greenberg. Interactive global illumination in dynamic scenes. *ACM Transactions on Graphics*, 21(3):537–546, July 2002. 5

74. J. Tumblin, J. K. Hodgins, and B. K. Guenter. Two methods for display of high contrast images. *ACM Transactions on Graphics*, 18(1):56–94, January 1999. 31

75. J. Tumblin and H. E. Rushmeier. Tone reproduction for realistic images. *IEEE Computer Graphics & Applications*, 13(6):42–48, November 1993. 27, 29

76. V. Volevich, K. Myszkowski, A. Khodulev, and E. A. Kopylov. Using the visual differences predictor to improve performance of progressive global illumination computations. *ACM Transactions on Graphics*, 19(2):122–161, April 2000. 24

77. G. Ward. A contrast-based scalefactor for luminance display. In *Graphics Gems IV*, pages 415–421. 1994. 29

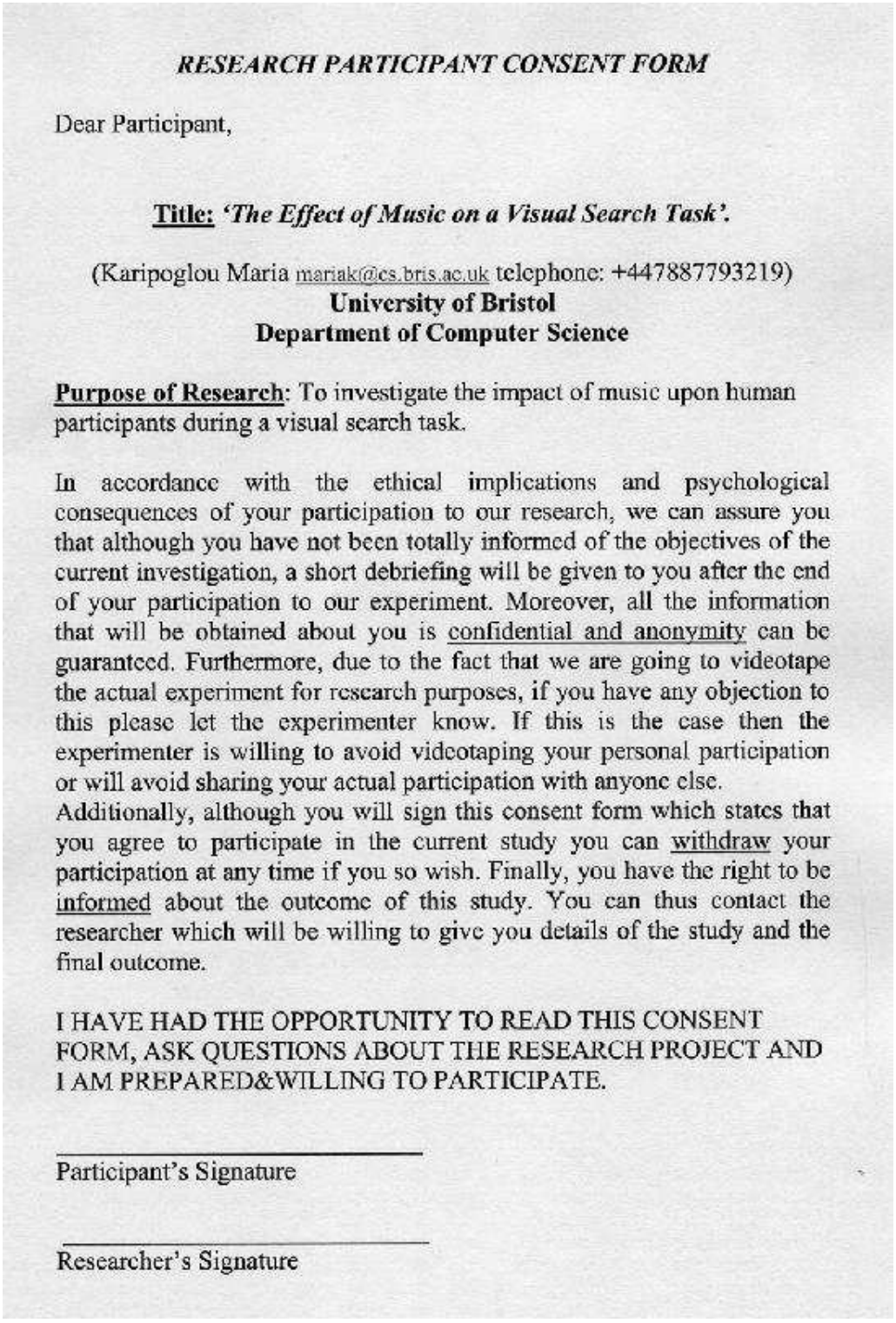
78. G. Ward and R. Shakespeare. *Rendering with Radiance: The Art and Science of Lighting Visualisation*. Morgan Kaufmann Publication, 1997. 27, 30

79. L. Williams. Casting curved shadows on curved surfaces. In *Computer Graphics (Proceedings of SIGGRAPH 78)*, volume 12:3, pages 270–274, August 1978. 5

80. A. Woo, P. Poulin, and A. Fournier. A survey of shadow algorithms. *IEEE Computer Graphics and Applications*, 10(6):13–32, November 1990. 4

81. Y. Yu, P. E. Debevec, J. Malik, and T. Hawkins. Inverse global illumination: Recovering reflectance models of real scenes from photographs. In *Proceedings of SIGGRAPH 99*, Computer Graphics Proceedings, Annual Conference Series, pages 215–224, Los Angeles, California, August 1999. 9, 34

82. Y. Yu, A. Ferencz, and J. Malik. Extracting objects from range and radiance images. *IEEE Transactions on Visualization and Computer Graphics*, 7(4):351–364, October 2001. 4



***RESEARCH PARTICIPANT CONSENT FORM***

Dear Participant,

**Title:** ***'The Effect of Music on a Visual Search Task'.***

(Karipoglou Maria mariak@cs.bris.ac.uk telephone: +447887793219)
**University of Bristol**
**Department of Computer Science**

**Purpose of Research**: To investigate the impact of music upon human participants during a visual search task.

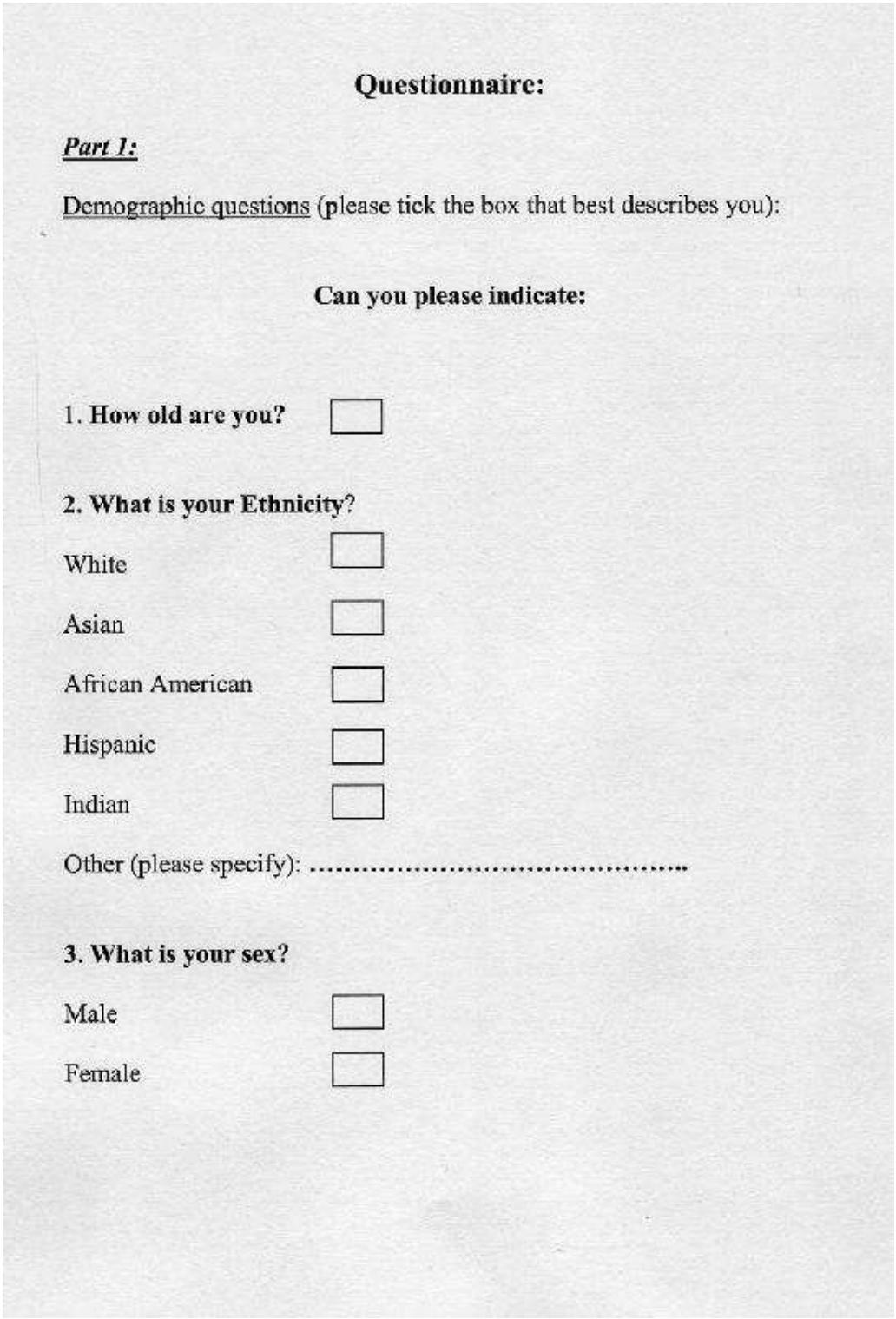
In accordance with the ethical implications and psychological consequences of your participation to our research, we can assure you that although you have not been totally informed of the objectives of the current investigation, a short debriefing will be given to you after the end of your participation to our experiment. Moreover, all the information that will be obtained about you is confidential and anonymity can be guaranteed. Furthermore, due to the fact that we are going to videotape the actual experiment for research purposes, if you have any objection to this please let the experimenter know. If this is the case then the experimenter is willing to avoid videotaping your personal participation or will avoid sharing your actual participation with anyone else.
Additionally, although you will sign this consent form which states that you agree to participate in the current study you can withdraw your participation at any time if you so wish. Finally, you have the right to be informed about the outcome of this study. You can thus contact the researcher which will be willing to give you details of the study and the final outcome.

I HAVE HAD THE OPPORTUNITY TO READ THIS CONSENT FORM, ASK QUESTIONS ABOUT THE RESEARCH PROJECT AND I AM PREPARED&WILLING TO PARTICIPATE.

_____
Participant's Signature

_____
Researcher's Signature

**Figure 29:** *Research participant consent form.*



**Questionnaire:**

***Part 1:***

Demographic questions (please tick the box that best describes you):

**Can you please indicate:**

1. **How old are you?** ☐

**2. What is your Ethnicity?**

White ☐

Asian ☐

African American ☐

Hispanic ☐

Indian ☐

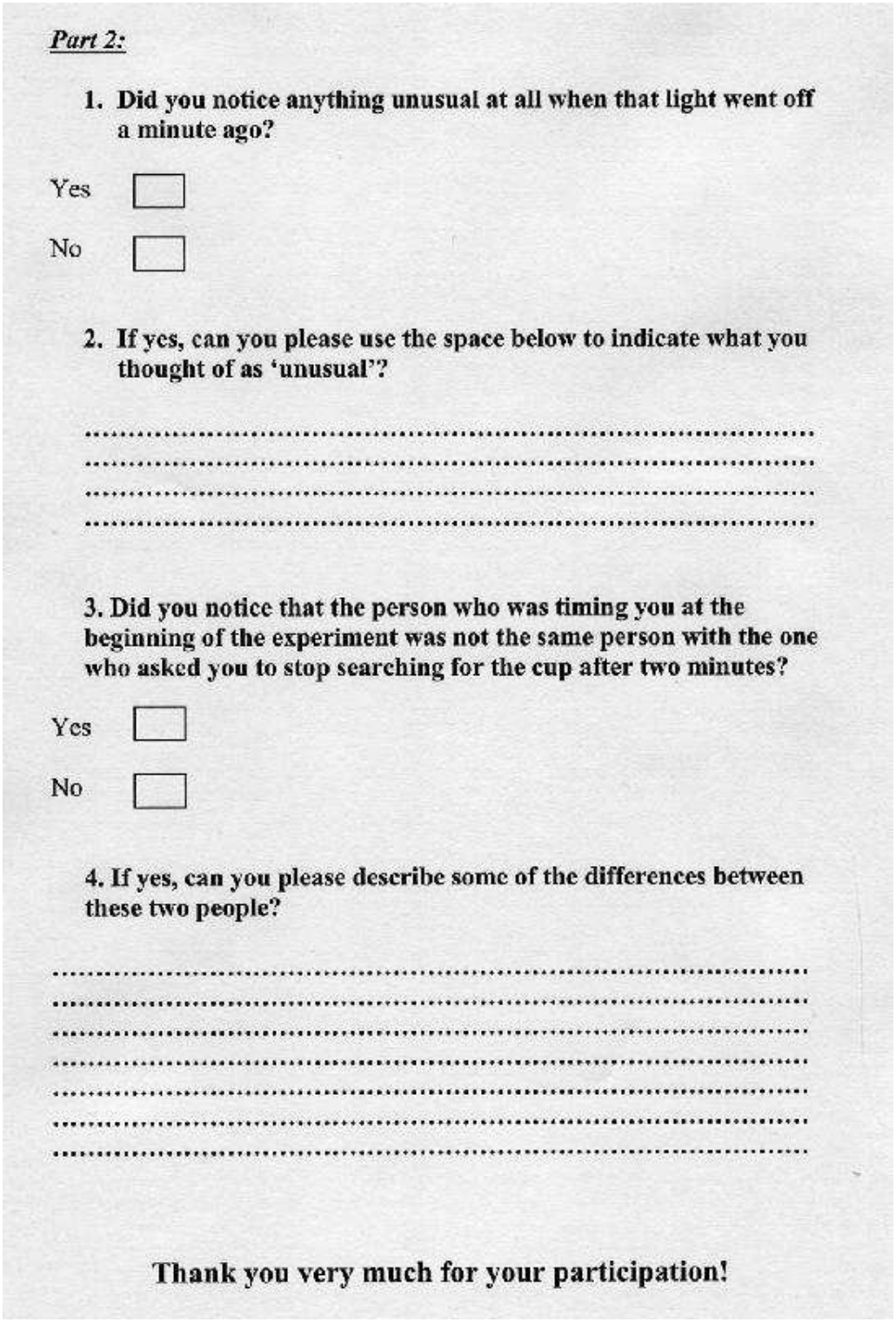
Other (please specify): .....

**3. What is your sex?**

Male ☐

Female ☐

**Figure 30:** *An example questionnaire, used in a study of inattentional blindness*[46].



***Part 2:***

1. **Did you notice anything unusual at all when that light went off a minute ago?**

Yes ☐

No ☐

2. **If yes, can you please use the space below to indicate what you thought of as 'unusual'?**

.....
.....
.....
.....

3. **Did you notice that the person who was timing you at the beginning of the experiment was not the same person with the one who asked you to stop searching for the cup after two minutes?**

Yes ☐

No ☐

4. **If yes, can you please describe some of the differences between these two people?**

.....
.....
.....
.....
.....
.....
.....

**Thank you very much for your participation!**

**Figure 31:** *Continued from Figure 30...*