Rapid Interactive Modelling from Video with Graph Cuts

Anton van den Hengel[1], Anthony Dick[1], Thorsten Thormählen[1], Ben Ward[1], and Philip H. S. Torr[2]

[1]School of Computer Science, University of Adelaide, Australia
[2]Department of Computing, Oxford Brookes University, UK

Abstract

We present a method for generating a parameterised model of a scene from a set of images. The method is novel in that it uses information from several sources—video, sparse 3D points and user input—to fit models to a scene. The user drives the process by providing selected high-level scene information, for instance selecting an object in the scene, or specifying the relationship between a pair of objects. The system combines this information with image and 3D data to dynamically update its model of the scene. In doing so it avoids common pitfalls of both automatic structure and motion algorithms, and image-based modelling packages.

Categories and Subject Descriptors (according to ACM CCS): I.4.8 [Image Processing and Computer Vision]: Scene Analysis

1. Introduction

Building 3D models of scenes from image data has been the subject of significant research effort and commercial interest. Structure-and-motion techniques for recovering point-based scene reconstructions and camera parameters are becoming well understood, to the point where commercial applications offer the technology (such as boujou from 2d3, amongst others). These techniques automatically generate a 3D point cloud, and an understanding of the relationship between the camera and the scene. The camera information is particularly useful for a number of video manipulation processes, including the insertion of computer generated elements into real video. The 3D point cloud, however, is a sparse reconstruction of the scene structure which can be difficult to interpret and use for modelling purposes.

There are also a number of systems which facilitate the modelling of scenes from image data; these include Facade [TDM96], Photobuilder [RC00], and Canoma from MetaCreations. These systems create models which provide a more complete and semantic reconstruction of the scene. A building might be modelled as a set of cuboids sitting on a plane, for example. This type of model facilitates a number of processes that would be difficult or impossible with a point cloud. It is possible, for example, to infer information about parts of the scene not visible in the image set, to remove objects, or to improve image-based rendering results. These modelling programs require significant user input, as the majority of their calculations are based on information obtained by user interaction.

We present here a method which combines the benefits of these two types of systems. The method uses structure-and-motion estimation to generate an initial point-based reconstruction and the associated camera information. This in turn informs the interactive modelling of the scene, removing most of the burden of object specification from the user. Importantly, however, the system still allows full user control if necessary. By relieving the user of burden of specifying information which may be acquired automatically the method allows the rapid creation of large and detailed models of real world scenes.

In Section 2, we describe briefly the framework that underlies the system. The remainder of the paper examines the process of interacting with the system to model a scene, and what is happening “behind the scenes”. After introducing the interface in Section 3, we describe in Section 4 how a single object of a particular type is selected in an image, without the need for precise localisation. In case this process fails, the option remains for the user to interactively add more information about the model as described in Section 5. We then progress to interactions that involve more than one object. In Section 6, we show how to interactively specify that a pair of objects is adjacent. We then show in Section 7 how the repetition of an object can be indicated very simply, with the details being estimated by the system.

2. Overview

We aim to find the set of 3D models $\mathtt{M} = \{M_i \ : i = 1 \ldots N\}$ that are most probable given the data $\mathtt{D}$ (images, camera parameters and 3D points) and any prior information $\mathtt{I}$. Models are specified by a vector of parameters—for example, a cube is represented by a scale S, position $\mathbf{T}$ and orientation $\mathbf{R}$—and can be seen as a function mapping those parameters to vertex locations. Models can be related to each other and these dependencies are expressed probabilistically through their parameters.

We represent the estimation problem as a Markov Random Field with a hidden node corresponding to each object in the scene and an observed node for each object observation. Hidden nodes are added to capture the relationships between objects. Observed nodes are linked to the corresponding model nodes, as would be expected, and relationship nodes are linked to the model nodes they affect. The Hammersley-Clifford theorem states that we can factorise the joint probability over the model set $\mathtt{M}$ as the (normalised) product of the individual clique potential functions of the graph formed by the nodes and their links [Bes74]. The cliques in this case are all of size 2.

The potential function adopted for the cliques containing an observed node and a model node is a product of model likelihood and prior, as:

$$\Pr(M|\mathtt{DI}) \propto \Pr(\mathtt{D}|M\mathtt{I}) \Pr(M|\mathtt{I}) \,. \tag{1}$$

The potential function for cliques representing inter-object relationships is the joint probability $\Pr(M,R)$ of the model M and the relationship R. Examples of this will be given later.

The joint probability of the set of models $\mathtt{M}$ and the set of relationships $\mathtt{R}$ given the data set $\mathtt{D}$ and the prior information $\mathtt{I}$ is thus

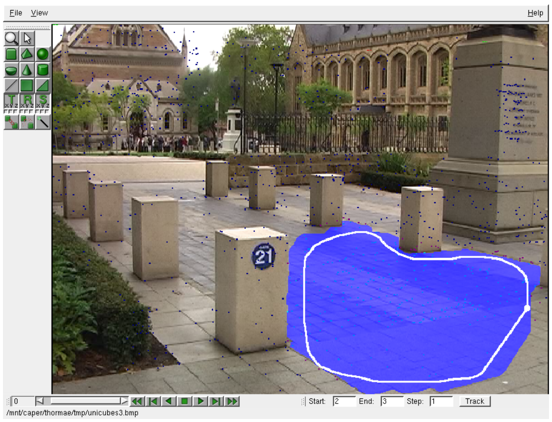
$$\Pr(\mathtt{M}|\mathtt{DI}) = \frac{1}{Z} \prod_{M \in \mathtt{M}} \Pr(\mathtt{D}|M\mathtt{I}) \Pr(M|\mathtt{I}) \prod_{R \in \mathtt{R}_M} \Pr(M,R), \tag{2}$$

where the set $\mathtt{R}_M$ represents the set of object-group relationships involving M, and the scalar Z a constant chosen such that $\Pr(\mathtt{M}|\mathtt{DI})$ integrates to 1.

Our goal is to find $\mathtt{M}$ that maximises the joint probability, or, equivalently, that minimises the negative log joint probability (i.e. the sum of the negative logs of the terms on the right hand side of Equation (2)). Each term in the joint probability is informed by user interaction and optimised by the system. In the rest of the paper we describe the system and how it accepts user input to assist its search for the most probable model.

3. The user interface

The interface to the system is much like that of other image-based modelling systems. The user is presented with one of the input images within a GUI which allows the selection of one of a small number of object types. In contrast to other systems, however, the image is overlaid with the projections of the reconstructed points, which provides an estimate of the depth of each region in the image.



0

/mnt/caper/thormae/tmp/unicubes3.bmp

Start: 2 End: 3 Step: 1

Figure 1: *The User interface, showing an initial object identification (a plane) and the resulting segmentation.*

4. Selecting an object

The user initiates the fitting process by specifying a model type, and highlighting the projection of the object to be modelled in one of the images. This requires that the user select a particular image from the input image set, and then simply draw a freehand line on the object of interest. The system then closes the curve if necessary. An example of this can be seen in Figure 1.

As the curve is drawn, the system attempts to segment the object of interest from the image. The appearance of the object is represented by a histogram of pixel values from within the currently selected region. This is compared with another histogram of pixel values from outside the region, by computing their Kullback-Leibler divergence. The segmentation we seek is that which maximises the divergence between the selected object and the background. The segmentation is optimised using the graph cuts method for solving dynamic Markov Random Fields developed in [KT05]. The efficiency of the method allows the segmentation to be updated quickly after the user completes the interaction. Part of this efficiency is achieved by initialising subsequent segmentations with the solution generated from the previous interaction. The speed with which these updates occur means that the user can start with quite a crude approximation and refine it only as far as is required.

As camera parameters and some 3D points are available, the region identified by the user and subsequently segmented can be used to determine an initial estimate of the parameters of the 3D object that projects into that area. In the case of the example in Figure 1 the model is a plane, and the plane parameters are estimated from the set of reconstructed points which project into the segmented area. The negative log likelihood of the plane is proportional the sum of squared distances between reconstructed points and the plane surface. Thus by minimising this distance we maximise the likelihood term in Equation 2. Other models are fitted in the same way, using the point to surface distance as a likelihood measure. This fit is further refined by reprojection into the image, as described in [vdHDT*06].

In contrast to other systems the user does not need to specify parallel lines in the plane, its boundaries, or any other information. The previous application of the structure and motion process means that simply identifying the area in one of the images is sufficient.

5. Refining an object

Once an object has been selected, and automatically segmented from the rest of the image, the user can refine the segmentation and the resulting 3D object if necessary. To do this, the user picks one or more vertices of the model and moves them to effect changes to the 3D shape of the model. The effects of the changes are shown in both a 3D and a 2D view, with the image superimposed. Because the system is aware of the type of model, interaction can be tailored to that model. For example, by picking one vertex of a sphere, the radius of the entire sphere can be changed. An example of this interaction is shown in Figure 2.

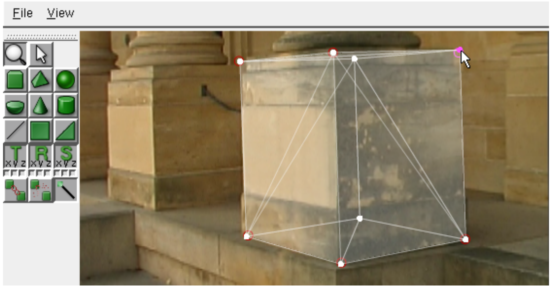


Figure 2: *Manual adjustment of vertex locations in one of the input images by dragging in the image.*

Any adjustment made by the user is incorporated probabilistically into the model fit by updating the prior term $\Pr(M|\mathtt{I})$ in Equation 2. For example, if the user moves a shape to a particular position in 3D, the prior on the model's position parameter becomes a Gaussian distribution centred about that location. The full joint probability is then reoptimised taking into account this prior.

6. Inter-object relationships

Selection of subsequent objects can be constrained not only by drawing on an image but also by specifying the relation of new objects to existing objects in the scene. A common case is adjacency—for example, in the scene shown in Figure 1, we can specify that a bollard rests on the ground plane we have already highlighted. To specify this relationship, the user first selects the bollard as described in Section 4. The bollard and the plane are then both selected, and a toolbox button pressed to indicate their adjacency. If the pillar is identified in isolation, there are insufficient reconstructed points to estimate the parameters of the model with the required accuracy. Once this relationship is incorporated, however, the extra constraint means that initial modelling can be carried out successfully.

When the user specifies a relationship between two objects the system incorporates this probabilistically into its estimate of each model. In this case the term $\Pr(M,R)$ is updated to become a Gaussian distribution centred at object parameters such that vertices from each object are coincident. The choice of which vertices are coincident is made using some prior knowledge that is encoded with each model. For instance, a cuboid is likely to be resting on one of its 6 faces. Thus there are 6 possible combinations to be tested.

7. Object repetition

It is common in urban and man-made environments to observe repeated instances of an object. Thus the ability to fit multiple copies of a model to a scene simultaneously is a powerful modelling facility. To fit multiple copies of an object, the user selects the object, and then simply drags the mouse along the direction in the image in which repetition occurs. Wireframe models are then superimposed along this line, in positions whose appearance most resembles the original object. The number and spacing of the models is determined automatically; however the user can override this manually if necessary. This is done by scrolling the mouse wheel to increase or decrease the model count.

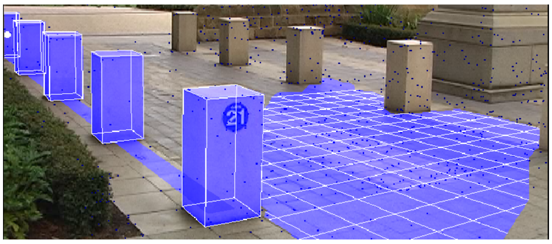


Figure 3: *Replicating multiple instances of a single object.*

Appearance similarity is once again measured by comparing histograms of pixel colours. The number of model instances is determined by the number of equally spaced locations that can be found along the indicated direction whose appearance is sufficiently similar to the original object. Only pixels belonging to the most visible face of the object are included in the histogram—this makes the system more robust to occlusion, lighting and other artifacts associated with viewing a face very obliquely. Object repetition is another case of an inter-object relationship and is thus incorporated into the model fit via the term $\Pr(M,R)$. The probabilistic nature of the relation allows some deviation from exact regularity.

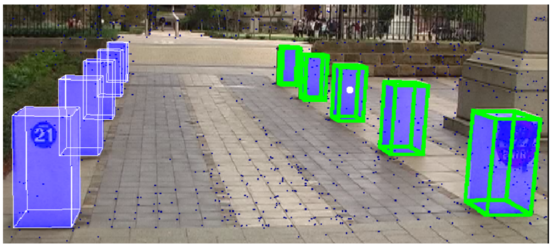


Figure 4: *Replicating a group of objects.*

8. Results

We present a number of results achieved using the method. Figure 5 shows an architectural scene with an overlay based on the reprojected estimated scene model. The reprojection into each of the original images shows that the model is accurate, and inspection of the model itself shows that it corresponds well to the true scene shape. The model of each pillar is made up of 3 stacked cuboids. The front pillar was modelled interactively with the remaining pillars modelled by replicating this front pillar. The pillars are not equally spaced along the line, but the spacing is regular enough for the replication process to have automatically selected the appropriate number of model instances. The final fit reflects the true spacing of the pillars despite being initialised with regularly spaced models.

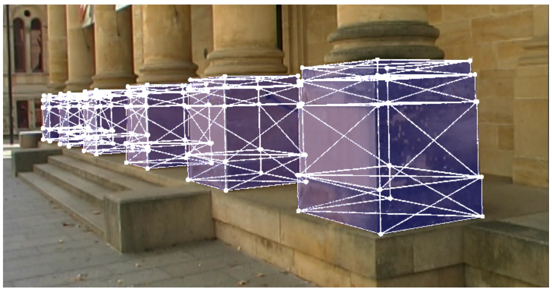


Figure 5: *Compound object replication*

Figure 6 shows an image which has had computer generated objects inserted on the basis of the model developed in Figures 1, 3 and 4. The shadow, for instance is projected onto the recovered plane and the pins are rendered so as to appear to be sitting on the recovered cuboid models.

Figure 7 shows selected frames from an input image sequence depicting a set of cubes on a table. The cubes are not regularly arranged, but abut the same plane. They have thus been identified individually by the user using the graph cut segmentation based closed curve method described in Section 4. Figure 7 also shows the same input frames modified on the basis of the recovered scene model. The modification has replaced each identified cube with a rendered model of a Rubik's cube.

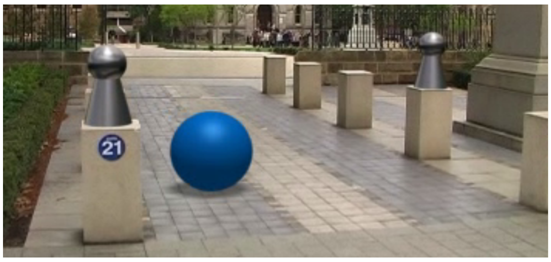


Figure 6: *A frame from a video showing computer generated objects inserted on the basis of a recovered scene model.*

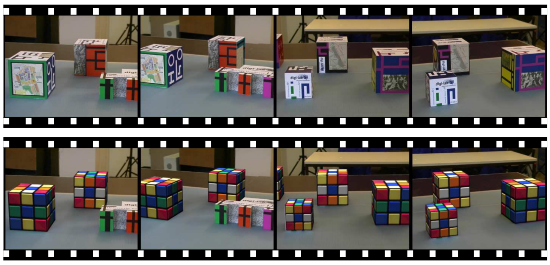


Figure 7: *Part of an input video sequence and the corresponding set of images modified according to a recovered scene model.*

References

[Bes74] Besag J.: Spatial interaction and statistical analysis of lattice systems. *Journal of the Royal Statistical Society. Series B (Methodological) 32*, 2 (1974), 192–236.

[KT05] Kohli P., Torr P.: Efficiently solving dynamic markov random fields using graph cuts. In *Proceedings of Tenth IEEE International Conference on Computer Vision* (October 2005).

[RC00] Robertson D., Cipolla R.: An interactive system for constraint-based modelling. In *Proc. 11th British Machine Vision Conference* (2000), pp. 536–545.

[TDM96] Taylor C., Debevec P., Malik J.: Modeling and rendering architecture from photographs: A hybrid geometry- and image-based approach. *ACM SIGGraph, Computer Graphics* (1996), 11–20.

[vdHDT*06] van den Hengel A., Dick A., Thormaehlen T., Torr P. H. S., Ward B.: Fitting multiple models to multiple images with minimal user interaction. In *Proc. International Workshop on the Representation and use of Prior Knowledge in Vision (WRUPKV), in conjunction with ECCV'06* (May 2006), (to appear).