

Synthetic Ground Truth for Feature Trackers

Johan Hedborg and Per-Erik Forssén
Computer Vision Laboratory
Department of Electrical Engineering
Linköping University
{hedborg,perfo}@isy.liu.se.

Abstract

Good data sets for evaluation of computer vision algorithms are important for the continued progress of the field. There exist good evaluation sets for many applications, but there are others for which good evaluation sets are harder to come by. One such example is feature tracking, where there is an obvious difficulty in the collection of data. Good evaluation data is important both for comparisons of different algorithms, and to detect weaknesses in a specific method.

All image data is a result of light interacting with its environment. These interactions are so well modelled in rendering software that sometimes not even the sharpest human eye can tell the difference between reality and simulation. In this paper we thus propose to use a high quality rendering system to create evaluation data for sparse point correspondence trackers.

1 Introduction

Structure from motion algorithms can gain quite a bit of stability by increasing the precision of the feature tracker used. Sub-pixel accuracy, and little or no bias, are important properties when deciding on what kind of tracker to use. To evaluate the performance of a feature tracker at this accuracy level is a very difficult task.

To stabilize the structure from motion calculations, two main tracker qualities are especially important. To have as little bias as possible in the measurements and to keep track of the feature as long as possible in the image sequence.

The modeling capacity of a modern 3D rendering engine is simply huge and has the possibility to produce an output that sometimes is indistinguishable from a real photo. We propose the use of such a system to generate complex images, that

can be used as evaluation data in the case of structure from motion. The light models used are very complex and can sufficiently well simulate many of the phenomena that cause problems and decrease performance of a feature tracker of today. The flexibility of these rendering systems will be a great asset when trying to locate weaknesses in current trackers. The optical flow for these sequences can easily be collected with near machine precision.

2 Related Work

A common way to evaluate the performance of a feature tracker is to use manual inspection. A set of points is tracked for a number of frames and then manually compared to their original location in the scene to see if the tracker has managed to track the point correctly, see e.g. [3]. This method is simple and effective, but there are however some drawbacks to it. One drawback is that it makes it hard to determine the bias and sub pixel accuracy of the tracking result. The result will also be subjective due to the human factor.

Higher precision can be achieved if an industrial robot arm is used. The camera is then mounted on the end effector of the robot. By moving the camera in a known trajectory in a static world with a known 3D structure it's possible to retrieve accurate point trajectories in the image sequence. One obvious limitation with this method is the limited range of the camera movement. Longer sequences with large scale differences are hard to collect, it would require a macro lens and miniature set up of some kind. There are also quite a few calibration steps involved, and the costs of such a setup are very high.

The people behind the Middlebury stereo evaluation set have developed an new evaluation set for optical flow [1]. There are three main data sets. The first is based on real world data and

are produced in a quite interesting way. Different objects are sprayed with fluorescent paint of different colors. Photos are taken with a high resolution camera while the objects are illuminated with ultra violet light. Photos are also taken in normal lightning condition, where the fluorescent paint is barely visible. The point correspondence is found using a SSD (Sum of Square Differences) tracker on the fluorescent images. Then every 20th of the normal lighted images are down sampled and it is they that are evaluation images. There are a couple of drawbacks when collecting evaluation data in this matter. It is not possible to capture scenes of larger environments due to spray painting and UV-lightning problems. Camera motion is restricted because of the need for a number of high resolution images in-between every evaluation frame. The data set needs to be recorded slower than real time to be able to capture all the images.

The second data set is a realistic synthetic set of images. The scenes was generated using Mental Ray [2], a high quality render engine. They are rendered with complex shapes, such as trees, in an outdoor environment. The scene also contains 3D texture simulation, small bumps, and details on the surface.

Finally, the third set is an adoption of there stereo data for rigid scenes. This allows for a comparison between the optical flow algorithms and the state-of-the-art stereo algorithms. The stereo scene structure and depth are collected using structured light and laser range imaging. The Middlebury evaluation sets are made solely for the purpose of evaluating dense optical flow. Because of this the test sequences are very short, each set consists of just 8 frames. The purpose of our evaluation data is to evaluate the performance when tracking more sparsely located points for a longer period of time. In our data set there are also the 3D models of the synthetic data, meaning that we have an exact 3D position of all possible landmarks in the sequence. This gives us the possibility to not only measure the performance of the tracker but also measure the performance of the structure-from-motion part.

2.1 Modeling Capabilities

Research in computer graphics has been ongoing for a long period of time. Not just at the universities but also in the industry, where large companies such as Pixar and others have helped raise the bar of computer generated graphics. The picture quality archived by a modern offline render engine today is very high, making the result sometimes in-

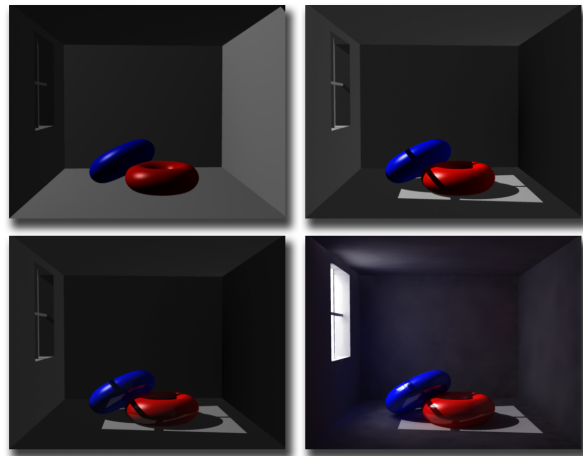


Figure 1: Different lightning models, u. left: simple shading, u. right: shadows added, l. left: added reflection on the toruses. l. right: Global Illumination.

distinguishable from a real photo. Many of these advances in image qualities comes from more realistic models of light, material and That is why we believe that with synthetic generated data we can reproduce many of the phenomena that decrease the performance of today’s feature trackers.

There are different models for simulating a shadow. The most realistic and accurate of them is the raytraced soft shadow A raytraced soft shadow extends the simple model where all light from a light sources comes from a single point of a light source to a model where the light emerges from an area. This gives us as the name indicated a much softer edge of the shadow then the normal raytraced shadow.

Almost all materials have some reflective components. The reflective properties of materials can make them look quite different. There is the mirror with almost perfect reflection. But there can also be other reflective surfaces such as brushed metal which has a reflective surface that is very rugged and gives a completely different look in a photo. Or a stone of granite that reflects light very differently across the surface. Small parts of the stone have almost total reflectivity while other parts are dark diffuse and reflect very little light. This can all be simulated by adjusting the reflective properties of a material.

A standard modeling assumption that is frequently used when rendering an image is that all light that hits a surface comes from a given light source. This assumption is pretty harsh because in the real world every visible surface reflects light.

The color of a small surface area in a room does not only depend on all the light sources in the room but also on the reflected light of walls and objects in the room. There are a number of algorithms that deal with these global reflectivity phenomena of objects and they go under the general name Global illumination. They are often very computationally heavy but can be rendered in a reasonable time on a modern processor.

Camera and lens characteristics give other photographic effects. When photographing a scene there will be some light that bounces around in the lens of the camera. These photographic side defects are called lens flares. When the contrast of lights in the scene is large these effects can be very evident.

Motion blur arises from fast motion or long exposure times. By rendering multiple samples of the models in the scene and averaging temporally, this effect can be simulated with high quality. Depth of field can be simulated with the same multiple samples technique.

When combining all the methods above to render a computer generated image, the result can become very realistically looking.

2.2 Generation of the Ground Truth Motion Field

The *motion field* is the field of projected 3D motion vectors onto the image plane. The *optical flow* is the *apparent motion* of brightness patterns in the image [1]. When we generate ground truth using a 3D model, we obtain a ground truth motion field, and not an optical flow field.

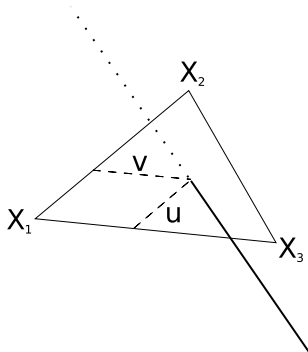


Figure 2: Intersection of cast ray and triangle.

When rendering a 3D model on a computer, the model is almost always divided up into small triangles. The corners of a triangle \mathbf{X}_1 , \mathbf{X}_2 , and \mathbf{X}_3 are called vertices. Each vertex is associated with certain properties i.e. color, surface normal

and more. When vertex properties differ within the same triangle these values need to be interpolated, to achieve smoothness where it's needed. This is usually achieved automatically by graphics hardware. We can use this mechanism to obtain the flow field between two images, and this only takes a few lines of shader code.

For each image coordinate $\mathbf{p} = (p_1 \ p_2)^T$, we cast a ray, and determine the closest intersecting triangle, and the (u, v) coordinates in that triangle. These uniquely specify the intersection point \mathbf{X} as:

$$\mathbf{X} = \mathbf{X}_1 + u(\mathbf{X}_2 - \mathbf{X}_1) + v(\mathbf{X}_3 - \mathbf{X}_1). \quad (1)$$

See figure 2. The three vertices are projected into the two cameras as:

$$\mathbf{x}_k = \mathbf{P}_1 \mathbf{X}_k \quad \text{and} \quad \mathbf{x}'_k = \mathbf{P}_2 \mathbf{X}'_k. \quad (2)$$

Here $\mathbf{x} = (x_1 \ x_2 \ x_3)^T$ denotes a vector in homogeneous coordinates, and $\tilde{x} = (x_1/x_3 \ x_2/x_3)^T$ is the corresponding coordinate where the projective scale has been normalized out. In order to determine the projection of the intersection point, we make use of the retained scale in the projected vertices \mathbf{x}_k , and combine them using a linear interpolation:

$$\mathbf{x} = \mathbf{x}_1 + u(\mathbf{x}_2 - \mathbf{x}_1) + v(\mathbf{x}_3 - \mathbf{x}_1). \quad (3)$$

The same procedure is then repeated in the second frame, to obtain \mathbf{x}' . The final motion field vector is then given as:

$$\mathbf{d} = \tilde{\mathbf{x}} - \tilde{\mathbf{x}}'. \quad (4)$$

An example of two rendered frames, and the induced ground truth motion field is given in figure 3.

2.3 Detection of occluded points

The procedure described in the previous section will give us a motion field vector $\mathbf{d}(\mathbf{p})$ in each image location, \mathbf{p} . However, for some of the points, it may happen that they are no longer visible in the second camera, either due to self occlusion, or because the triangle has been occluded by another object. If the ground truth is going to be useable, we need to detect these points.

One way of detecting occluded points is to render images where each pixel contains the index of the triangle visible in that pixel. We denote these *triangle index images* by T_1 and T_2 for frames 1 and 2 respectively.

Now we loop through the generated motion field image \mathbf{d} , and for each location \mathbf{p} , we check

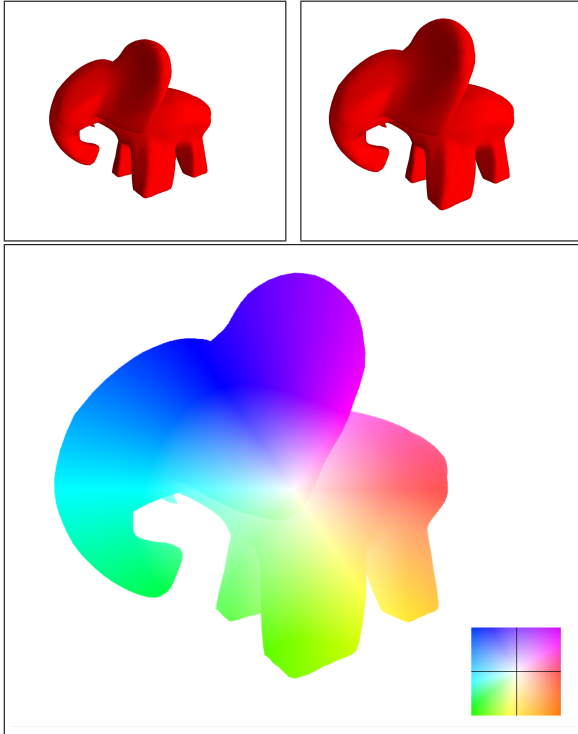


Figure 3: Rendering example. Top: frame 0 and frame 1. Bottom: The induced motion field from frame 0 to frame 1. Motion vectors are colour coded according to the coordinate system in the lower left corner.

whether the triangle image T_1 contains the same value as any of the corresponding positions in T_2 :

$$M(\mathbf{p}) = \begin{cases} 1 & \text{if } T_1(\mathbf{p}) = T_2(\mathbf{q}_1) \\ 1 & \text{if } T_1(\mathbf{p}) = T_2(\mathbf{q}_2) \\ 1 & \text{if } T_1(\mathbf{p}) = T_2(\mathbf{q}_3) \\ 1 & \text{if } T_1(\mathbf{p}) = T_2(\mathbf{q}_4) \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

Here $\mathbf{q}_1 \dots \mathbf{q}_4$ are the four nearest neighbours to the floating point position $\mathbf{p} + \mathbf{d}(\mathbf{p})$. We need to consider all four neighbours, in order not to detect also the pixels that are less than half a pixel away from a triangle edge.

3 Further work

One of the real strengths of synthetically made evaluation data is the ability to generate data with large variety of different settings for different kinds of optical effects. We can get quantitative measurements of the performance of a tracker when different optical effects are present or not. We

can maybe isolate specific problems that cause the tracker to perform badly.

The most commonly used tracker today is based on planar rectangular features and a pure translation model. If more degrees of freedom are released they often lose their robustness. With high quality synthetically rendered data there is a possibility to generate ground truth data with a large variety of settings. These variations can be difference in light settings, different geometrical properties, a variety of material properties and more. The data can be seen as a large set of training examples, and the tracker could be trained to handle these variations. Doing this we could hopefully loosen some more degrees of freedom for the tracker with maintained robustness. A possible future goal is to have the tracker self adjust according to the tracking circumstances.

4 Acknowledgments

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement no 215078.

References

- [1] S. Baker, D. Scharstein, J. P. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. In *IEEE International Conference on Computer Vision*, volume CFP07198-CDR, Rio de Janeiro, Brazil, October 2007. IEEE Computer Society.
- [2] T. Driemeyer. *Rendering with mental ray*. Springer-Verlag New York, Inc, 2001.
- [3] H. Wu, R. Chellappa, A. C. Sankaranarayanan, and S. K. Zhou. Robust visual tracking using the time-reversibility constraint. In *IEEE International Conference on Computer Vision*, volume CFP07198-CDR, Rio de Janeiro, Brazil, October 2007. IEEE Computer Society.