Lessons and Insights from Creating a Synthetic Optical Flow Benchmark

Jonas Wulff[1], Daniel J. Butler[2], Garrett B. Stanley[3], and Michael J. Black[1]

[1] Max-Planck Institute for Intelligent Systems, Tübingen, Germany
{jonas.wulff,black}@tuebingen.mpg.de
[2] University of Washington, Seattle, WA, USA
djbutler@cs.washington.edu
[3] Georgia Institute of Technology, Atlanta, GA, USA
garrett.stanley@bme.gatech.edu

Abstract. With the MPI-Sintel Flow dataset, we introduce a naturalistic dataset for optical flow evaluation derived from the open source CGI movie Sintel. In contrast to the well-known Middlebury dataset, the MPI-Sintel Flow dataset contains longer and more varied sequences with image degradations such as motion blur, defocus blur, and atmospheric effects. Animators use a variety of techniques that produce pleasing images but make the raw animation data inappropriate for computer vision applications if used "out of the box". Several changes to the rendering software and animation files were necessary in order to produce data for flow evaluation and similar changes are likely for future efforts to construct a scientific dataset from an animated film. Here we distill our experience with Sintel into a set of best practices for using computer animation to generate scientific data for vision research.

1 Introduction

In recent years standardized datasets have been a driving force in various subfields of of computer vision such as object recognition and optical flow [1]. The benefits to the field are twofold: First, they make algorithms objectively comparable and, by creating challenges, inject a sense of competition into the community. Second, by providing a large corpus of data, ideally representative of the world, they make it possible to use learning-based techniques.

Today the Middlebury flow dataset [1] remains the most widely used optical flow benchmark. However, current top methods perform similarly on this dataset, making it hard to determine which differences in performance are significant. In this sense, the Middlebury flow dataset may no longer be challenging enough to drive continued innovation in the field.

In order to provide a more challenging optical flow benchmark, we have developed a new dataset based on the open source CGI movie Sintel[1], which we call MPI-Sintel [2]. Compared to Middlebury, it contains larger motions, non-rigidly

[1] http://www.sintel.org

A. Fusiello et al. (Eds.): ECCV 2012 Ws/Demos, Part II, LNCS 7584, pp. 168–177, 2012.

Fig. 1. Examples. Example frames from 8 of the 35 scenes contained in our dataset. The ground truth optical flow is shown below the corresponding rendered frame. Color coding as in [1].

moving objects, and image degradations such as motion and defocus blur. Figure 1 shows 8 example scenes from our dataset and the associated ground truth optical flow. This paper serves as a companion to the primary publication introducing the dataset and benchmark [2]. Our main contributions in this paper are 1) to describe how state-of-the-art rendering software can be used to generate a high-quality synthetic optical flow dataset; 2) to point out the pitfalls that arise when using existing 3D data to generate such a dataset as well as possible solutions; and 3) to draw general conclusions relevant for users of synthetic data for computer vision applications.

1.1 The Value of Synthetic Datasets

To train and evaluate optical flow algorithms, one would ideally use a large number of natural video sequences and the corresponding ground truth optical flow. Unfortunately, no sensor currently exists that can measure optical flow directly. In some cases, one can compute the flow from auxiliary information such as special texture patterns [1] or range imagery [3,4]. Hidden texture patterns, however, can only be used in laboratory environments with controlled illumination [1]. Optical flow maps generated from depth and camera motion suffer from occlusion artifacts, noisy depth data, and interpolation error. In [4], for example, only about 50% of the pixels contain usable ground truth optical flow. Furthermore, depth-based methods only work for rigid scenes.

One of the goals of optical flow estimation is to deal with arbitrary scenes and non-rigid motions. To circumvent the problems described, we use a synthetic (computer-generated) dataset. In this case the scene geometry and its change over time are known so one can trace each pixel $\mathbf{p_t}$ from the image plane to its origin point $\mathbf{P_t}$ in the scene, track that point to its location $\mathbf{P_{t+1}}$ in the next frame, and compute the projection $\mathbf{p_{t+1}}$ in the next frame. Up to machine precision, the ground truth optical flow is then given by $\mathbf{u_t} = \mathbf{p_{t+1}} - \mathbf{p_t}$.

In addition to high accuracy, using computer-generated scenes has two further benefits. First, even realistic-looking scenes are relatively easy to create in large quantities, which is important for methods that use machine learning techniques. Second, the creator of the scene potentially has total control over all aspects of the data generation, from the raw geometry to the image rendering process. Thus, one can change aspects of the scene, such as the materials, the light, or the motion of the camera or the objects, and generate multiple renderings of the same scene using varying parameters. This allows one to evaluate the impact of individual parameters on the performance of optical flow algorithms.

The main argument against synthetic data is that the rendered images may differ from images of the real world in important ways. This critique certainly applies to previous synthetic flow benchmarks shown in Fig. 2. However, the realism and complexity of CGI that is possible today makes synthetic data worth reconsidering. Feature films, for example, often use graphics that is indistinguishable from real scenes and seamlessly combine real and CGI elements.

The dataset we propose here is derived from the CGI movie Sintel, a 14-minute short movie. It was created and rendered using the rendering software Blender[2]. Both Blender and the complete production data for Sintel are released as open source. The availability of the Blender's source code allowed us to understand and control how exactly the images were generated, something that would not have been possible using a proprietary rendering software. Furthermore, since the production data is freely available, we can modify, extend, and re-generate all parts of the movie.

2 Previous Data Sets

Datasets for optical flow evaluation can be subdivided into three categories: *natural*, containing relatively unaltered photographic images, *semi-synthetic*, containing either heavily processed or elaborately staged images, and *synthetic*, containing completely computer-generated images. Since the dataset we present here is completely synthetic, we limit this section to synthetic datasets. A review of datasets of natural and semi-synthetic datasets is given in [1].

Barron et al. [5] developed the first synthetic dataset for optical flow evaluation, which included the widely used Yosemite sequence. Yosemite is still part of the Middlebury dataset [1] but contains only grayscale images, rigid motion, no atmospheric effects such as haze, no shading, and few occlusions.

McCane et al. [6] proposed a synthetic dataset containing varying degrees of scene complexity, object motion, and camera motion, with only the highest level of scene complexity resembling a real scene (see Fig. 2). Compared to the Yosemite sequence, it contains color images and separately moving objects, but only very simple textures.

The Middlebury dataset [1], today's most widely used optical flow evaluation dataset, introduced new synthetic sequences, Urban1-3 and Grove1-3, resembling a city and natural outdoor scenes, respectively. Compared to the complex scenes

[2] `http://www.blender.org`

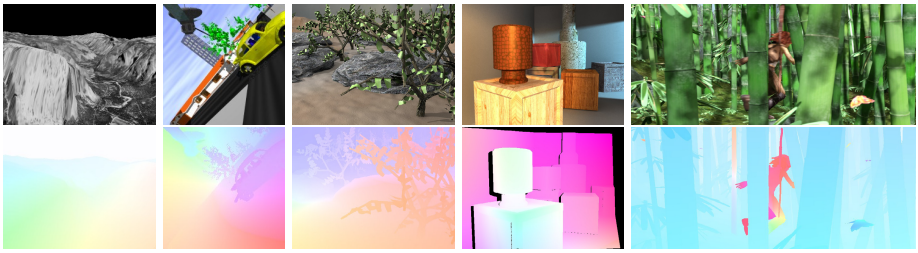

Fig. 2. History: Evolution of synthetic flow datasets. Example frames are shown on top, the corresponding color coded optical flow at the bottom. From left to right: Yosemite sequence from [5]; Complex sequence from [6]; Grove3 sequence from [1]; 009_Crates1 sequence from [7]; Bamboo3 sequence from MPI-Sintel [2].

from [6], these sequences contain lighting effects such as shadows, rich textures, and more complex object structure. They do not, though, contain atmospheric effects or image degradations, and display only very little independent object motion and mostly small velocities.

UCL-Flow [7] introduces 20 new image pairs with strong rigid object motion and more complex light situations with colored light sources and ambient lighting. Its intended use is to train a mechanism to locally select the best optical flow algorithm, given a set of image features.

Whether synthetic datasets are in fact realistic enough to predict performance on real images remains an open question. Vaudrey et al. [8] compare performance on synthetic and semi-synthetic sequences from Middlebury to performance on a real-world dataset, recorded from a driving car, and find that the results are worse on the natural scenes. The main differences between the synthetic Middlebury sequences and these real sequences are motion blur and violations of brightness constancy.

Meister and Kondermann [9] re-create a real scene in graphics software enabling them to compare performance on real and CGI sequences. They find that, while the locations of the errors are somewhat different, the average errors for the synthetic sequence and the real sequences are similar when using advanced rendering techniques such as global illumination and carefully selected textures.

In summary, existing synthetic datasets fall short on a number of dimensions, making them less complex than real video sequences: the rendering quality is often unconvincing, motions are limited to rigid objects and very small velocities, no degradations such as motion and focus blur are included, and the available data is often limited to image pairs.

3 Generating Synthetic Data

When generating synthetic flow data, two things are important: The *production data* that describes the 3D shape, appearance, and motion of the scenes, and the *rendering method* used to transform the production data into actual video

sequences. Below we describe both components and the modifications required to make the output usable as an optical flow evaluation dataset.

3.1 Production Data

All production data (geometry, textures, compositing setups etc.) for Sintel is available as open source. We used the latest version of the data[3], which contains all the data and software used to create the original Sintel film.

Like most CGI-based films created for entertainment, Sintel contains a number of effects that help achieve a certain look, but pose problems for the scientific use of the data. In order to make the Sintel data usable for an optical flow evaluation data set, a number of changes had to be made. These changes can be subdivided into two categories: changes due to limitations of current optical flow representations, and changes due to problems with the data and/or Blender.

Changes due to Limitations of Optical Flow Representations. Optical flow is usually represented as a two-dimensional, pixel-based map, with velocities in x and y direction at each pixel. Since each pixel can only have a single motion, the complexity of the scene is limited. A transparent surface, for example, cannot be realistically captured, since a pixel cannot be assigned both the motion of the surface and the motion of the background visible through it. A similar problem arises when an object (such as a single hair) is smaller than a pixel, or when object boundaries are anti-aliased. In this case, a single pixel contains multiple, potentially different velocities. The intensity value of such a pixel might be a weighted combination of the foreground and background. In an optical flow representation, though, such a blending does not make sense, since it would assign a pixel the average velocity between the foreground and the background.

Therefore, we disable all transparencies of objects such as glass or feathering of hair, and render the optical flow without anti-aliasing. The optical flow value for a pixel is then the motion at its center. Additionally all strands of hair are set to be rendered at least 2 pixels wide. While this makes the hair appear somewhat "cartoon-y", it ensures correct optical flow at every pixel that predominantly displays hair. Fig. 3 shows an example of a scene before and after these changes.

Changes Due to Problems with the Data. Several problems were caused by bugs within Blender or by artistic "shortcuts", which go unnoticed in the movie but introduce artifacts in the still images or the optical flow fields.

Scene-wise hair problems. Rendering hair as described above effectively broke the relatively complex hair shading system[4], resulting in hair that was often lit inconsistently with respect to the rest of the scene – sometimes too bright, sometimes too dark. We solved this problem by manually adjusting the hair shading for each scene to match the ambient lighting.

[3] Available at `http://download.blender.org/durian/svn/`

[4] For more details on hair shading in Sintel, and the resulting difficulties, see `wiki.blender.org/index.php/Org:Institute/Openprojects/Durian/HairNotes`

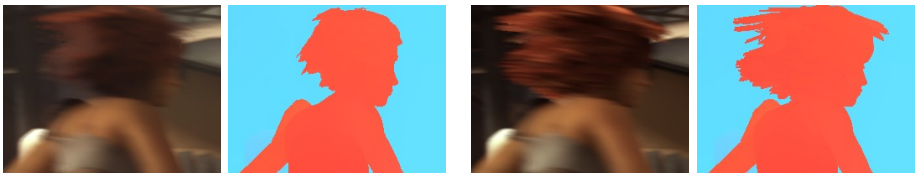

Fig. 3. Illustration of the hair transparency issue. Comparison of the rendered images and optical flow maps in the original form (left) and with our changes applied (right). Due to transparency effects, a large amount of hair is not captured in the optical flow of the original data.

Motion blur. The motion blur in Sintel is simulated using the optical flow field. The choice of the number of samples used in the blur filter represents a trade-off between accuracy and computing time. We found that in most of the scenes, it was set to a relatively low value, sometimes leading to visible artifacts. To fix this, we set the number of samples to 1024.

Texture problems. In Blender, displacement maps (i.e. textures that deform an object based on their intensity) are integrated into the scene *after* the object motion is computed. For this reason, the computed optical flow field contains the motion of the undeformed object, leading to slight inaccuracies for objects with such displacement maps. The displacement component for all textures was therefore disabled. We find that this step does not significantly impact the data, both visually and in terms of optical flow accuracy. It should be noted, though, that *bump maps* were retained, since they do not alter the rendered vertex positions, but only the normal directions. Thus, textural structure is preserved.

Animation adjustments. In Blender, it is possible to animate a change in speed of a cyclic animation (for example, to let a bird flap faster or slower). However, due to a bug in Blender, this change of motion speed is not incorporated into the optical flow field, introducing errors. We solve this by setting the speed of all cyclic animations to a constant speed.

Threading. Certain settings in the global illumination computation in Blender lead to visible tiling artifacts if different parts of the images are rendered by different threads. Since our dataset consists of a large number of scenes, we can parallelize the rendering on a (scene-)level by computing all scenes in parallel. Without a loss in performance, we can then disable multithreading for each single Blender instance, thus eliminating the tiling problem.

Halos. For dramatic effect, some scenes include relatively large, transparent halos. Besides the transparency problems described above, we found that the presence of these halos causes problems in Blender's image generation pipeline, rendering the output images unusable. Therefore, halo effects were disabled.

3.2 Generation Process

With the changes described above, the images were rendered using Blender's internal raytracing engine. It supports a number of advanced techniques, such

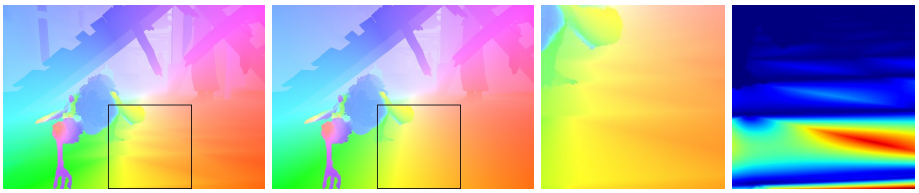

Fig. 4. Illustration of the optical flow interpolation problem. From left to right: The original optical flow map, containing triangulation artifacts; the same optical flow map, rendered with our modifications; a magnification of the marked region of the original rendering; the endpoint error in the marked region (difference between the correct 3D flow and the original Blender 2D flow).

as flexible material and specularity shaders, effects such as subsurface scattering, and global illumination, all of which are used heavily in Sintel. To create realistic looking images, the lighting is especially important. Blender uses ambient occlusion (causing a darkening of cavities), environment lighting (ambient light based on a sky texture), and indirect lighting. These illumination effects are computed using a Quasi-Monte-Carlo raytracing method, randomly sampling rays while trying to keep the spacing between the rays as even as possible. While this provides a good trade-off between accuracy, image quality, and computing time, it can introduce noise artifacts, as described in Sec. 4.1. Furthermore, it should be noted that, while these effects increase the perceived quality and realism of a scene, they only approximate real-world lighting.

A typical Sintel scene consists of a number of layers, defined by the artist. These layers correspond roughly to background/static and foreground/animated parts of the scene. Each layer is rendered separately and then combined with the other layers using a given compositing layout. The compositing layout defines the post-processing stages (e.g. color correction) for each layer, and how they are combined (for example based on their transparency or depth values) to form the final rendering. Since the compositing layout is mainly used for image post-processing, it cannot be used to combine the data (optical flow and depth). For example, a non-linear color correction would result in a meaningless distortion of the optical flow. When rendering the optical flow and depth, we thus collapse the whole scene onto a single layer, and render the data from that layer.

Blender computes optical flow as part of its rendering process for the purpose of simulating motion blur. Unfortunately, the computed flow is wrong, since for efficiency reasons, the computation is performed in 2D. Each vertex is projected onto the image plane in two subsequent frames, resulting in the true flow at these locations. However, the motion of pixels *inside* a triangle in between three vertices is given by linearly interpolating the *2D motions of the vertices*. The resulting inaccuracies become noticeably large in case of triangles slanted in depth. An example is shown in Fig. 4. To fix this, we modified Blender itself, and integrated a true 3D interpolation of the optical flow that is projected onto the image plane after the interpolation. Fig. 4 shows an example.

Using our modified version of Blender, the data was rendered into OpenEXR files. These files contain different layers: the data, such as the depth and flow information, as floating point values, and the images from the individual render passes as linear values. The images are then gamma-corrected using $\gamma = 0.4641$, the power Blender uses to generate standard image data (such as PNG files).

3.3 Data Selection

Without producer credits, the full movie has a length of approximately 18,000 frames, each of which has a size of about 7 MB (including depth and optical flow), amounting to a total data volume of 123 GB. We felt this was too large to be practical and use a subset of the movie containing 1628 frames from 35 scenes, chosen based on the following criteria:

Rendering consistency. Despite the efforts described above, some sequences remained problematic. In a few sequences, the depth ordering of the layers in the composite layout was inconsistent with the actual scene geometry. This leads to an object being in front of another in the images, but behind that same object in the flow and depth data, which are generated from the scene collapsed onto a single layer. Scenes where this posed a problem were excluded from the dataset.

Variability. Since the ultimate goal of this dataset is to evaluate optical flow algorithms, we wanted the contained data to be as varied and interesting as possible. We therefore exclude scenes that are mostly static, or that have similar optical flow to scenes already in the dataset.

Difficulty. One motivation for this dataset is to be more challenging than Middlebury. However, we also do not want it to be impossible to get good results. Therefore, scenes that are too easy (i.e. contain constant or very small motion) or too hard (scenes for which a human can not identify what is happening, based on an isolated, two-second snippet) are discarded.

4 Description of the Dataset

The final MPI-Sintel dataset contains 35 scenes, between 20 and 50 frames long, for a total of 1628 frames. The rendered images have a resolution of 1024 x 436 pixels at 24 frames per second. Extending previous synthetic datasets, it contains atmospheric effects, motion and defocus blur, and large velocities. The scenes span a large variety of environments, characters, and actions.

To investigate when optical flow algorithms break, we render each frame in three different passes: the *Albedo* pass contains no shading and satisfies brightness constancy everywhere except in occlusion regions, the *Clean* pass, which contains shading, but no image degradations, and the *Final* pass, which additionally includes motion blur, defocus blur, and atmospheric effects, and corresponds to the final movie. Fig. 5 shows examples of the different passes.

The dataset is split into a 1064 frame training set and a 564 frame test set[5]. The training and test sets are balanced in terms of first-order statistics of the

[5] All data can be obtained from `http://sintel.is.tuebingen.mpg.de`

Fig. 5. Render passes. Two examples for the different passes contained in the proposed dataset. From left to right: Albedo pass, Clean pass, Final pass

images and optical flow maps, and their respective spatial and temporal derivatives. For the training set, we provide ground truth optical flow, depth information, masks of invalid pixels, extrinsic and intrinsic camera data, maps of occluded regions, and motion boundaries. For the test set, this data is withheld for evaluation. Fig. 1 shows frames from 8 example scenes and their corresponding optical flow maps. A detailed investigation of the realism of the dataset, as well as the performance of optical flow algorithm, can be found in [2].

4.1 Remaining Problems

We took great care to eliminate problems with the data as much as possible. However, a few issues remain. The most obvious of these are occasional interpenetrations of objects, for example the main character's hair going through her collar. While comparable situations can in theory appear in the real world (for example, if a sharp object cuts through a piece of fabric), one expects these to appear far less often than in our dataset. We would advise against learning a physics-based world model from the data we present here.

In some cases, specular, highly structured textures contain too much high-frequency information for the anti-aliasing to work properly, causing a "sparkling" appearance. A similar effect with a different cause is a faint low-frequency noise in some textures. This is caused by the Global Illumination algorithm, which uses a limited, random sampling of light rays bouncing back from surfaces. Since the random sampling changes with each frame, the surfaces are effectively illuminated by a noisy light source. Both of these effects are quite subtle, and we believe that optical flow algorithms should be robust against them.

Another rendering-related issue is visible when an object of single pixel width is motion-blurred. Since foreground and background optical flow cannot be meaningfully blended, the flow is aliased. When part of an object has a width of less than a pixel, it my not be assigned a foreground flow value since the background flow is dominant. This causes the affected area to not be motion-blurred resulting in thin objects that can be partially motion blurred and partially sharp. To eliminate this effect, we rendered the affected scenes with double the resolution, and downscaled the images afterwards. This mostly eliminated the effect and, where it remains, it is confined to a very small spatial area.

5 Conclusion

We described the detailed process of generating the MPI-Sintel optical flow dataset from the open source CGI movie Sintel. This required several modifications to the data and the rendering method to to make the data suitable for use in such a benchmark. Summarizing, we find 3 key issues in using synthetic data: **1. Physical implausibilities.** Synthetic movies are made to look good, not to be physically accurate. They only roughly approximate the real world and we think it is important to compare them statistically with natural scenes to validate how realistic they are. **2. Control.** To make data usable in a scientific environment, it is important to have full control over the production data as well as the mechanisms and software used to generate or transform this data. **3. Representation.** Optical flow is usually represented as a two-dimensional motion vector field, which cannot model multiple motions at the same point. This representation limits the possible contents and complexity of the dataset. Future work should address richer representations, and evaluation measures, to allow datasets to include more complex phenomena.

With these principles, and progress in photorealistic rendering, we believe that synthetic data can play an increasing role in computer vision.

Acknowledgments. We thank T. Roosendaal for his assistance. MJB and GBS were supported in part by NSF CRCNS Grant IIS-0904630.

References

1. Baker, S., Scharstein, D., Lewis, J.P., Roth, S., Black, M.J., Szeliski, R.: A Database and Evaluation Methodology for Optical Flow. Int. J. Comput. Vision 92, 1–31 (2011)
2. Butler, D.J., Wulff, J., Stanley, G.B., Black, M.J.: A Naturalistic Open Source Movie for Optical Flow Evaluation. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV 2012, Part VI. LNCS, vol. 7577, pp. 611–625. Springer, Heidelberg (2012)
3. Roth, S., Black, M.: On the spatial statistics of optical flow. In: ICCV 2005, vol. 1, pp. 42–49 (2005)
4. Geiger, A., Lenz, P., Urtasun, R.: Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite. In: CVPR 2012 (2012)
5. Barron, J.L., Fleet, D.J., Beauchemin, S.S.: Performance of optical flow techniques. Int. J. Comput. Vision 12, 43–77 (1994)
6. McCane, B., Novins, K., Crannitch, D., Galvin, B.: On benchmarking optical flow. Comput. Vis. Image Underst. 84, 126–143 (2001)
7. Mac Aodha, O., Humayun, A., Pollefeys, M., Brostow, G.J.: Learning a Confidence Measure for Optical Flow. IEEE Trans. Pattern Anal. Mach. Intell. 34 (to appear, 2012)
8. Vaudrey, T., Rabe, C., Klette, R., Milburn, J.: Differences between stereo and motion behavior on synthetic and real-world stereo sequences. In: 23rd International Conference of Image and Vision Computing New Zealand (IVCNZ 2008), pp. 1–6 (2008)
9. Meister, S., Kondermann, D.: Real versus realistically rendered scenes for optical flow evaluation. In: CEMT 2011, pp. 1–6. IEEE (2011)