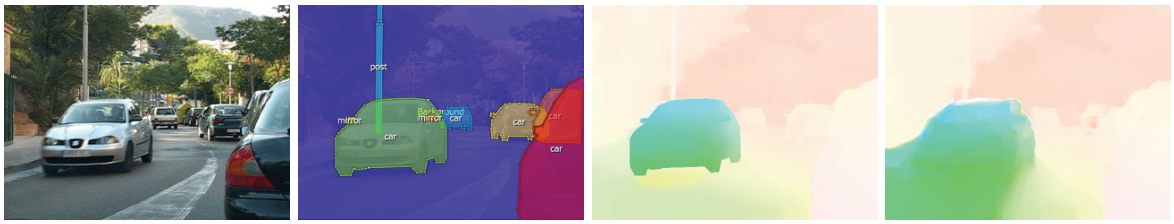
Human-Assisted Motion Annotation

Ce Liu[1] William T. Freeman[1] Edward H. Adelson[1] Yair Weiss[1,2]
[1] CSAIL MIT [2] The Hebrew University of Jerusalem
{celiu,billf,adelson}@csail.mit.edu yweiss@cs.huji.ac.il



(a) A frame of a video sequence (b) User-aided layer segmentation (c) User-annotated motion (d) Output of a flow algorithm [8]

Figure 1. We designed a system to allow the user to specify layer configurations and motion hints (b). Our system uses these hints to calculate a dense flow field for each layer. We show that the flow (c) is repeatable and accurate. (d): The output of a representative optical flow algorithm [8], trained on the Yosemite sequence, shows many differences from the labeled ground truth for this and other realistic sequences we have labeled. This indicates the value of our database for training and evaluating optical flow algorithms.

Abstract

Obtaining ground-truth motion for arbitrary, real-world video sequences is a challenging but important task for both algorithm evaluation and model design. Existing ground-truth databases are either synthetic, such as the Yosemite sequence, or limited to indoor, experimental setups, such as the database developed in [5]. We propose a human-in-loop methodology to create a ground-truth motion database for the videos taken with ordinary cameras in both indoor and outdoor scenes, using the fact that human beings are experts at segmenting objects and inspecting the match between two frames. We designed an interactive computer vision system to allow a user to efficiently annotate motion. Our methodology is cross-validated by showing that human annotated motion is repeatable, consistent across annotators, and close to the ground truth obtained by [5]. Using our system, we collected and annotated 10 indoor and outdoor real-world videos to form a ground-truth motion database. The source code, annotation tool and database is online for public evaluation and benchmarking.

1. Introduction

Motion analysis has been an essential component for many computer vision applications such as structure from motion, object tracking/recognition, and advanced video editing. A variety of computational models, *e.g.* optical flow fields [10, 15], layers [25] and boundaries [13], have been developed to estimate the motion from a video sequence. It is important to evaluate the performance of various motion analysis algorithms to gain insight and design better models.

However, little has been done for evaluating motion analysis algorithms compared to the tremendous effort put into developing these algorithms. For example, researchers have tried to optimize optical flow algorithms based on a synthetic sequence *Yosemite*, an overly simplified example compared to real-life videos [4]. As a result, the performance of the optical flow algorithms optimized for the Yosemite sequence may deteriorate significantly when sensor noise, motion blur, occlusion, shadow, reflection, auto white balance and compression artifacts occur in a sequence.

While it is important to have a ground-truth optical flow database consisting of real-world videos, annotating the motion for real-life videos can be challenging. Recently, Baker *et al*. designed a new database for optical flow evaluation [5]. Although the ground-truth flow of several sequences was measured, their methodology of painting fluorescent materials to track motion is limited to indoor scenes and artificial motion.

To address this problem, we propose a human-in-loop methodology to annotate ground-truth motion for arbitrary real-world videos. Our approach to obtaining motion ground truth is based on three observations. First, humans are experts at segmenting layers in a video sequence because we can easily recognize the moving objects and their relative depth relationships. Second, we are sensitive to any differences between two images when these two images are displayed back and forth. Third, humans have a knowledge of the smoothness and discontinuities of the motion that a moving object undergoes. In fact, computer vision researchers have implicitly used these observations to inspect the accuracy of motion estimates when the ground truth is missing, and even to verify the correctness of ground-truth annotation.

Labeling motion pixel by pixel and frame by frame can

978-1-4244-2243-2/08/$25.00 ©2008 IEEE

be tedious, and we designed a computer vision system to ease the labeling process. Typically, it takes four steps to label the motion of a video sequence with our system:

(a) **Semi-automatic layer segmentation**. The user specifies the contour and relative depth of an object in one frame and the system automatically tracks the contour for the rest frames with appropriate occlusion handing.

(b) **Automatic layer-wise flow estimation**. The user specifies a set of parameters for optical flow estimation for each of the layers, and selects the flow field that both produces the best visual match and agrees with the smoothness and discontinuity of a layer.

(c) **Semi-automatic motion labeling**. If the user is not satisfied with the flow estimation of a layer in step (b), the user can specify sparse correspondence and the system automatically fits parametric motion or interpolates a dense flow field until the user is satisfied with the motion.

(d) **Automatic compositing**. The system automatically composes the labeled motion of each layer into a full-frame flow field.

To evaluate our methodology, we provide quantitative evidence that human annotated motion is significantly closer to the real ground truth than any of the existing computer vision algorithms, and that human annotations made using our system are consistent across subjects.

Using our system, we have obtained and carefully labeled 10 video sequences with 341 total frames for both indoor and outdoor scenes (and we continue to label sequences). We also reported the performance of current optical flow algorithms on this database; we hope our labeled dataset will provide the feedback necessary to improve the state of the art for motion analysis in realistic video sequences. An additional byproduct of our system is ground-truth layer segmentation that can be used for evaluating layer segmentation and occlusion boundary detection algorithms. We make available the source code, labeling tool and database online for public evaluation and benchmarking `http://people.csail.mit.edu/celiu/motion/`.

2. Related Work

Modeling and estimating dense optical flow fields have been intensively studied in the literature. Starting with Horn & Schunck [10] and Lucas & Kanade [15], researchers have developed a variety of models for effective flow computation, including some recent work such as incorporating robustness functions [6], integrating gradient information [7], estimating a symmetric flow field [3], combining local and global flow [8], and reasoning about occluded/disoccluded pixels (outliers) [19]. The success of optical flow estimation has been shown on the Yosemite sequence, with the average angular error (AAE) within 2°, but this success does not reveal many examples where optical flow algorithms may fail. One motivation of our work is to have a ground-truth motion database with diverse examples to reveal these failures and understand the cause.

Researchers have argued that higher level representations such as layers and contours should be used to analyze motion. Since Adelson proposed the layer representation for videos [1], many computational models have been developed for automatic layer estimation [25, 26, 23, 27, 14]. Recently, preliminary success was obtained using a boundary representation for motion analysis [13]. Although most of the layer and boundary motion work focused on obtaining the correct segmentation, these higher level representations should also result in better full-frame motion estimation. Unfortunately, the goal of producing more accurate motion is missing from the existing work. We want to obtain a database to evaluate layer/boundary analysis algorithms, as well.

Closest to our goal is Baker *et al*.'s recent work [5], where the authors designed an elegant approach to obtaining ground truth optical flow. They painted fluorescent patterns onto objects in an indoor experimental setup and used a computer-controlled stage to generate motion. By taking two pictures simultaneously under ambient and UV light, they were able to capture both a natural image and an image with rich textures from the fluorescent patterns. A feature tracking system was designed to produce the ground-truth motion. Although the motion of a few sequences were successfully measured, their methodology is limited to controlled materials, motion and lighting conditions. In parallel, Roth and Black [17] obtained a synthetic ground-truth motion database using an existing depth database. In contrast, our approach is able to generate ground-truth motion for real-world videos under much broader imaging conditions.

Image annotation has been explored in the literature, such as the Berkeley image segmentation database [16] and LabelMe object annotation database [18]. These ground-truth databases have led to not only algorithm evaluation but also many other vision applications. One of our goals is to provide a broad ground-truth motion database for the computer vision community. Nevertheless, compared to these static image annotation tools which are relatively easy to design, designing a tool for motion annotation is nontrivial because accurate motion annotation and temporal consistency are required.

Interactive object segmentation in video sequences has recently been explored in computer graphics, including contour-based video rotoscoping [2], soft alpha matting [9], over-segmentation [24] and 3D graph cut [12]. We use the contour-based model [2] in our system because contours are intuitive to interact with. We allow the user to specify depth information so that contours can be tracked appropriately under occlusion. The goal of our system is to obtain both segmentation and motion, but our focus is motion, whereas these interactive segmentation tools focused on segmentation.

3. Human-Assisted Motion Annotation System

Accurately labeling every pixel in every frame of a video sequence is a nontrivial task, and we designed a computer vision based system to allow the user to annotate motion in a reasonable amount of time. We assume that a video sequence can be decomposed into layers, each of which has a binary mask and smooth flow field. There are three main components in our system: *semiautomatic layer segmentation*, *automatic optical flow estimation* and *semiautomatic motion labeling*. The model that we designed for each of the components will be explained in Sect 3.1 through 3.3. The human interaction and the graphical user interface will be briefly described in Sect 3.4.

We used the state-of-the-art computer vision algorithms to design our system. Many of the objective functions in contour tracking, flow estimation and flow interpolation have $L1$ norm for robustness concerns. Techniques such as iterative reweighted least squared (IRLS) [8, 7], pyramid-based coarse-to-fine search [6, 8] and occlusion/outlier detection [19] were intensively used for optimizing these nonlinear object functions.

3.1. Semiautomatic Layer Segmentation with Occlusion Handling

In this module, the user labels the contour of a moving layer in one frame, and the system automatically tracks the contour for the rest of frames. By allowing the user to specify a time-varying depth value for each layer, our system can handle contour tracking under occlusion. The user can correct the tracking, and the system automatically propagates the correction to other frames.

We feel that realtime performance is more important than accuracy for this user-interactive tracker. Therefore, we did not choose slow but accurate trackers such as particle filtering [11]. Our contour tracker is designed based on the models for optical flow estimation with occlusion handling incorporated.

Without loss of generality, suppose that the user defined a polygon using landmarks $\mathcal{L}=\{z_k:\, z_k \in \mathbb{R}^2\}_{k=1}^N$ at frame I_1. The task of contour tracking is to find the motion vector w_k for each landmark z_k at frame I_2. Notice that I_2 can be either after or before I_1 for forward or backward tracking. Intuitively, we want the contour to move consistently and to match local image features. The objective function is defined as:

$$E(\{w_k\}) = \sum_{k=1}^{N} \sum_{q \in N_k} r_k(q) |I_2(z_k + w_k + q) - I_1(z_k + q)| + \lambda \sum_{k=1}^{N} h_k |w_k - w_{k+1}|, \quad (1)$$

and $w_{N+1} = w_N$. In this equation, the weight h_k is used to take into account the distance between points z_k and z_{k+1}; we define $h_k = \frac{\overline{d}}{d_k + \overline{d}}$, where $d_k = \|z_k - z_{k+1}\|$, and $\overline{d}$ is the average of d_k. In this way, points that are closer are more likely to move together. Variable N_k is a square neighborhood centered at point z_k, and r_k is the region of support for z_k, a 2D Gaussian function modulated by a binary mask indicating whether each neighboring pixel q is inside the polygon.

The objective function in Eqn. (1) is nonlinear. Similar to many optical flow algorithms, we use the Taylor expansion [15] to linearize the data term, and optimize the objective function using coarse-to-fine search and IRLS. Images I_1 and I_2 contain not only RGB channels, but also the 1st- and 2nd-order derivatives of the luminance to account for lighting changes across frames. The coefficient λ controls the rigidity of the object: smaller λ indicates easier deformation for the object. We allow the user to adjust λ before tracking.

In order to handle occlusion, we allow the user to specify the relative depth value of a layer at some key frames, and the system automatically interpolates the depth (as a function of time) for the rest of the frames. The contour tracker is driven by a 2nd-order dynamical model for prediction, and the prediction is used as an initialization for optimizing Eqn. (1). The tracking algorithm iterates between the following two steps to handle occlusion:

(1) Check whether each landmark z_k is occluded by other layers with smaller depth values. If occlusion is detected for z_k, then set $r_k(q) = 0,\ \forall q \in N_k$ in Eqn. (1). This means there is no region to support tracking z_k.

(2) Optimize Eqn. (1) using the coarse-to-fine scheme.

This hypothesis-testing algorithm converges in a few steps.

Although this contour tracker works well for most objects we tested, contour tracking can drift, and can sometimes be wrong when the object rotates. The user can modify a landmark, and the system automatically propagates the correction to the rest frames. In this temporal propagation, the system estimates linear regression coefficients for the other points to reconstruct the point that the user modified, applies the same coefficients at other frames, and blends the reconstruction with the old value using a weight inversely proportional to the time difference. We found that this simple propagation algorithm works surprisingly well. In contrast, the sophisticated contour tracking/modification algorithm proposed in [2] is too expensive for realtime long-distance propagation.

Once contours are specified for each layer, the system calls a standard *inpolygon* routine to generate a binary layer mask, and uses the relative depth to generate a visible layer mask consisting of the layer pixels that are not occluded. Now the system will automatically generate dense correspondence, *i.e.*, an optical flow field for each layer at every frame.

3.2. Layer-wise Optical Flow Estimation

The key difference between layer-wise optical flow estimation and traditional full-frame optical flow estimation is a mask indicating the visibility of each layer. Only the pixels falling into this mask are used for matching. Because the shape of the mask can be arbitrary and fractal, outlier detection is performed to excluded occlusions in the flow computation.

We use the optical flow algorithms in [8, 7] as the baseline model for optical flow estimation, while symmetric flow computation [3] is also incorporated to improve the accuracy. Let M_1 and M_2 be the visible mask of a layer at frame I_1 and I_2, (u_1, v_1) be the flow field from I_1 to I_2, and (u_2, v_2) the flow field from I_2 to I_1. The objective function for estimating layer-wise optical flow consists of the following terms. First, a data term is designed to match the two images with the visible layer masks:

$$E^{(1)}_{\text{data}} = \int g * M_1(x,y) |I_1(x+u_1, y+v_1) - I_2(x,y)|, \quad (2)$$

where g is a Gaussian filter. The data term $E^{(2)}_{\text{data}}$ for (u_2, v_2) is defined similarly. Notice that $L1$ norm is used here to account for outliers in matching. Second, smoothness is imposed by

$$E^{(1)}_{\text{smooth}} = \int \left(|\nabla u_1|^2 + |\nabla v_1|^2\right)^{\eta}, \quad (3)$$

where η varies between 0.5 and 1. Lastly, symmetric matching is required:

$$E^{(1)}_{\text{sym}} = \int \begin{aligned} &\left|u_1(x,y) + u_2(x+u_1, y+v_1)\right| + \\ &\left|v_1(x,y) + v_2(x+u_1, y+v_1)\right|. \end{aligned} \quad (4)$$

The objective function is the sum of the above three terms:

$$E(u_1, v_1, u_2, v_2) = \sum_{i=1}^{2} E^{(i)}_{\text{data}} + \alpha E^{(i)}_{\text{smooth}} + \beta E^{(i)}_{\text{symtr}}. \quad (5)$$

We use IRLS, equivalent to the outer and inner fixed-point iterations proposed in [7], combined with coarse-to-fine search and image warping to optimize this objective function. After the flow computation at each pyramid level, we update the visible layer mask M_1 based on the estimated flow:

- If $M_2(x + u_1, y + v_1) = 0$, then set $M_1(x, y) = 0$;
- If the symmetry term in Eqn. (4) is beyond a threshold at (x, y), then set $M_1(x, y) = 0$.

The same rule is also applied to update M_2. As the algorithm runs from coarse to fine, we obtain both bidirectional flow fields and trimmed visible layer masks that reflect occlusion.

We allow the user to adjust α, β and η in Eqn. (5) for each layer to handle different elasticities. Intuitively, a larger α, β or η results in a smoother flow field, but it is smooth in different ways. The user typically does not know which parameter setting produces the best flow. Therefore, our system allows the user to specify a list of different parameter settings for each layer, and the system computes the dense optical flow field for each parameter setting of each layer at each frame. This computation is done offline.

Although our layer-wise optical flow estimation works well in general, we observe failures where no parameter setting generates reasonable flow. The failures are mainly caused by the following factors.

- Specularity, shadow, noise and blurriness in real-life videos cause the brightness or boundary preservation assumption to break.
- Due to occlusions, the visible layer mask may be very small or irregularly shaped, making the global motion difficult to capture.

In such cases when optical flow estimation fails, we rely on semiautomatic motion labeling.

3.3. Semiautomatic Motion Labeling

When optical flow estimation fails, the user can specify sparse correspondence between two frames using feature points, and our system automatically generates a parametric motion or interpolates a dense optical flow field based on the sparse correspondence. The sparse correspondence can be specified either with a computer's assistance for efficiency, or manually, to give the user full control of motion annotation.

When the user specifies one feature point in one frame, the system automatically searches for the best match in the next frame using mininum-SSD matching and the Lucas-Kanade transform [15] for sub-pixel accuracy. Based on the number of specified feature points, the system automatically determines the mode of parametric motion−translation, affine transform or homography−and estimates the motion parameters accordingly [22]. The user can also select these modes, and even choose to produce a smooth flow field interpolated using the preconditioned conjugate gradient algorithm described in [21].

However, specifying corner-like feature points [20] can be difficult for some sequences when only line structures are present in the layer. To address this problem, we incorporate uncertainty matching and probabilistic parametric motion estimation so that the user can freely choose any pixel for correspondence. In uncertainty matching, the system generates a probability map $p_k(x)$ for the matching of feature point k at location $q_k \in \mathbb{R}^2$. This probability map $p_k(x)$ is approximated by a mean μ_k and covariance matrix Σ_k. In probabilistic motion estimation, the system iterates between the following two steps. In the first step, motion

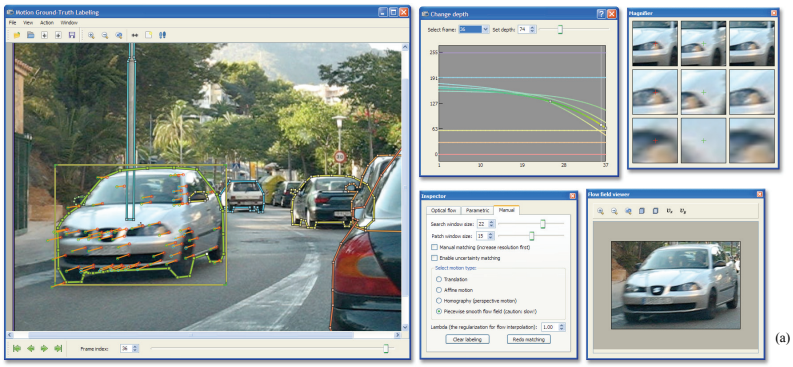


Figure 2. A screenshot of our motion annotation system. From (a) to (e) is the main window, depth controller, magnifier, flow field viewer and control panel.

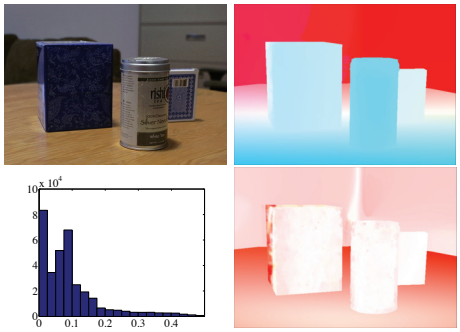


Figure 3. Clockwise from top left: the image frame, mean labeled motion, mean absolute error (red: high error, white: low error), and error histogram.

is estimated using the current estimate of mean and covariance. Mathematically, let $h(q_k;\theta):\mathbb{R}^2\mapsto\mathbb{R}^2$ be a parametric motion applied to q_k. The motion parameter is estimated by

$$\theta^* = \arg\min_{\theta} \sum_k \big(h(q_k;\theta)-\mu_k\big)^T \Sigma_k \big(h(q_k;\theta)-\mu_k\big). \quad (6)$$

In the second step, the mean and covariance are estimated using a new probability map reweighted by the current motion

$$\{\mu_k, \Sigma_k\} \leftarrow p_k(x)\mathcal{N}(h(q_k;\theta^*), \sigma^2\mathbf{I}). \quad (7)$$

This algorithm converges within a few iterations. The motion $h(q_k;\theta)$ can also be a dense optical flow field (*i.e.* θ). In addition, the feature points that the user labeled can be used for the next frame. To perform the semiautomatic motion labeling, the user interacts with these tools through an interface, described next.

3.4. System Design and Human Judgement

A graphical user interface (GUI) is a key component of our system since intensive user interaction is needed for motion annotation. Our system is developed in C++ with Qt$^{\text{TM}}$ 4.3 for the GUI. A screenshot of the GUI is shown in Figure 2. Our system allows the user to label contour and specify correspondence in the main window (a), change depth in a dialog window (b) and modify parameters in an inspector window (e). We also provide a magnifier to see image details (c) and a flow field viewer to check optical flow field and matching (d).

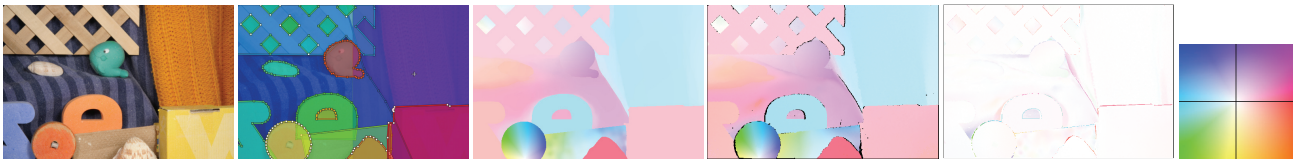
We found that two criteria must be satisfied for a computer-generated flow field, either from dense optical flow estimation or from sparse correspondence, to be accepted by a human as the right motion for a layer. First, the matching has to be correct. The flow field viewer, as shown in Figure 2 (d), is used to display the current frame and the warped next frame based on the flow field back and forth, to check whether this flow field produces the right match. Second, the smoothness and discontinuities of the flow field must match the rigidity and boundary of the object. This property can also be inspected in the flow field viewer.

The user interaction time depends on the number of frames, the number of layers and the shape of the objects. Typically, it takes 10 to 40 minutes to label five or six objects in a 30-frame sequence, depending on the shape complexity and the amount of occlusion. It takes several hours to compute the dense flow field for every flow parameter setting, every layer and every frame. It takes one minute or more to specify the sparse correspondence for a layer, and fitting parametric motion in realtime.

4. Methodology Evaluation

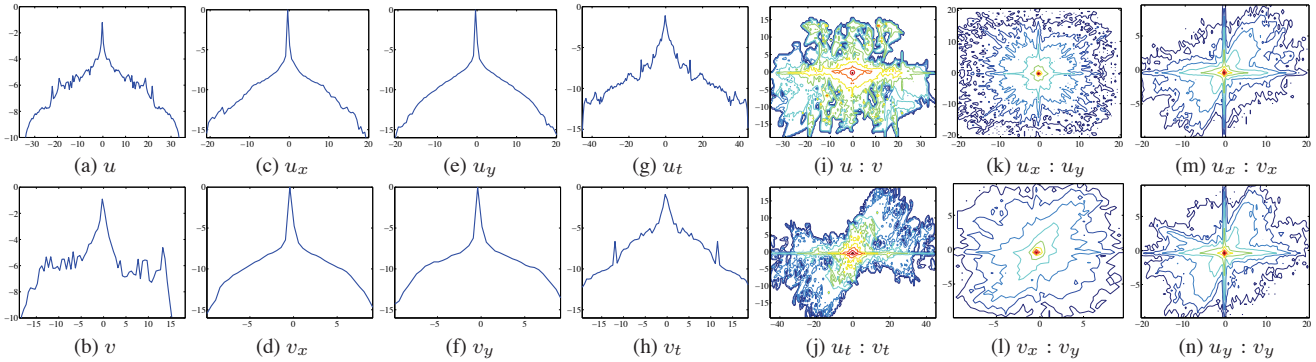
We conducted two experiments to examine our methodology. First, we applied our system to obtaining motion for the sequences that have been carefully annotated using other methods. We downloaded the ground-truth motion of a sequence RubberWhale from the webpage of [5]. We labeled the sequence using 20 layers, generated the layer-wise motion, and showed the results in Figure 4. Most of the disagreement lies along the occluding boundaries of the objects, as shown in Figure 4(e). The error between our annotation and their "ground-truth" flow is $3.21°$ in average angular error (AAE) and 0.104 in average endpoint error (AEP). For comparison, the best optical flow algorithm achieves merely $11.09°$ in AAE for a similar sequence Seashell [5]. Notice that the authors in [5] generated the flow on high-res images and then down-sampled the flow to produce the ground truth, whereas we directly worked on the low-res images.

Second, we tested the consistency of multiple users' labelings. We asked nine subjects to use our tool to annotate motion for the sequence in Figure 6 (a). Since the subjects agreed with the layer segmentation of the sequence, we labeled the layers and generated ten flow fields for each layer by varying the smoothness parameters. The subjects were asked to choose the best flow field for each layer, and they could label the sparse correspondence if they were not satisfied with any of the flow fields. We found some subjects



(a) A selected frame (b) Layer labeling (c) User-annotated motion (d) Ground-truth from [5] (e) Difference (f) Color map

Figure 4. For the RubberWhale sequence in [5], we labeled 20 layers in (b) and obtained the annotated motion in (c). The "ground-truth" motion from [5] is shown in (d). The error between (c) and (d) is 3.21° in AAE and 0.104 in AEP, excluding the outliers (black dots) in (d). (e): The color encoding scheme for flow visualization [5].



(a) u (c) u_x (e) u_y (g) u_t (i) $u : v$ (k) $u_x : u_y$ (m) $u_x : v_x$

(b) v (d) v_x (f) v_y (h) v_t (j) $u_t : v_t$ (l) $v_x : v_y$ (n) $u_y : v_y$

Figure 5. The marginal ((a)∼(h)) and joint ((i)∼(n)) statistics of the ground-truth optical flow in our database (log histogram).

preferred smooth flow fields over good matching, while others preferred good matching over smoothness, but all the subjects tried to balance between smoothness and correct matching. The subjects were unanimously unsatisfied with the flow field of the table, and all of them labeled the motion of the table using feature point matching.

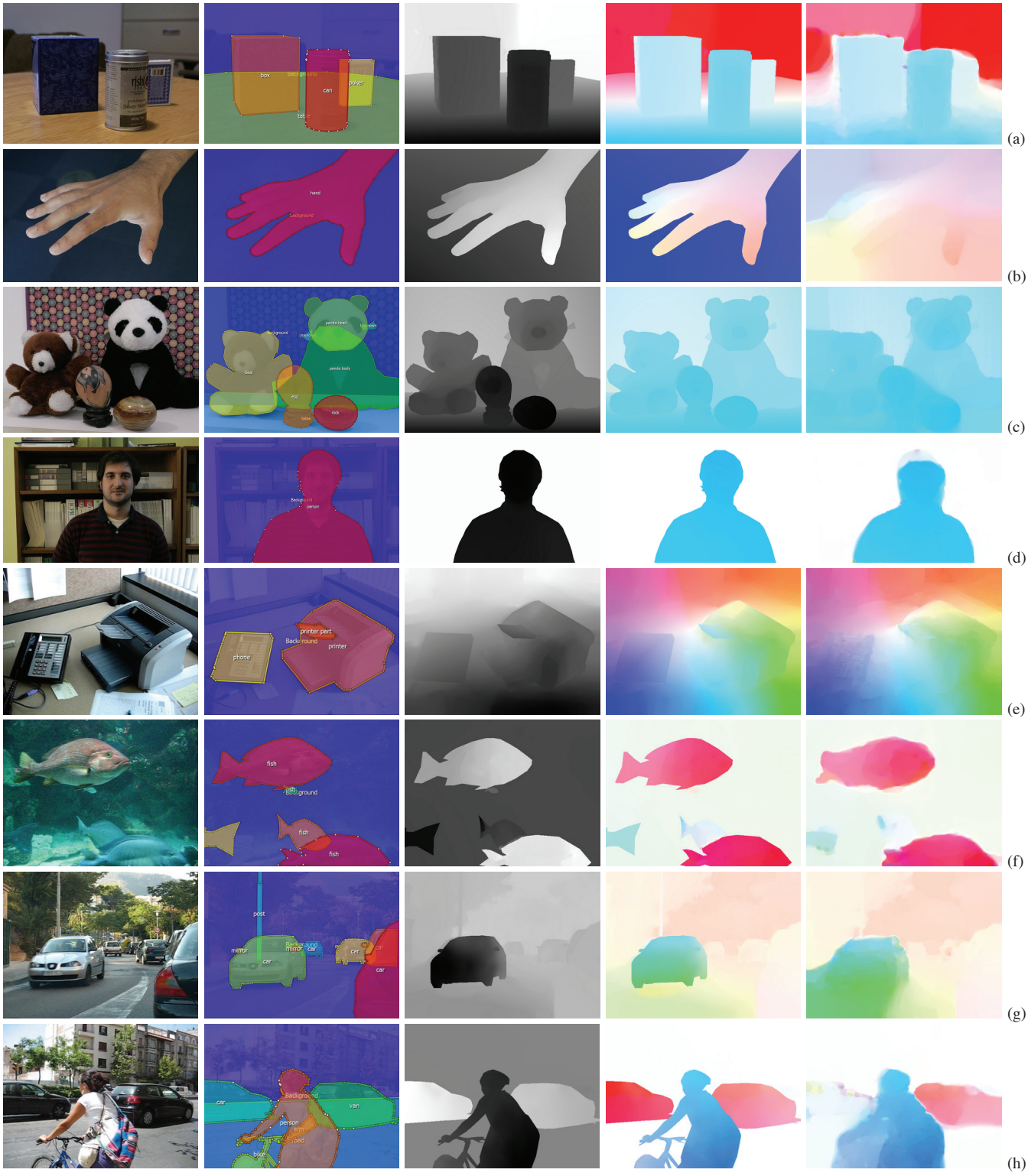
The per-pixel mean and standard deviation of these nine subjects' annotations are shown in Figure 3. The mean of the standard deviation is 0.0934 pixel. Most of the disagreement is at the table area, where some subjects labeled more than 10 points and some only labeled five. We also measured the error between each annotation and the mean flow field. The mean error is 0.989° in AAE and 0.112 in AEP. This AEP value is consistent with the previous experiment: the accuracy of human annotation is around 0.1 AEP.

5. A Human-Annotated Motion Ground-Truth Database

We collected video sequences of both indoor and outdoor scenes using a Canon EOS-1D (low frame rate, medium resolution) and a Canon SD870 (high frame rate, middle resolution), and carefully labeled the motion using our annotation tool. Some sequences captured by the Canon EOS-1D are shown in Figure 6 (a) to (d). We observe noisy and blurry backgrounds in (a) and (b) because of the shallow depth of field. Some of the sequences captured by the Canon SD870 are displayed in Figure 6 (e) to (h). A typical frame of the selected sequence is shown in column (1), the corresponding layer labeling in column (2) and horizontal motion in column (3). To compare the annotated flow field and the result of a state-of-the-art flow algorithm [8], we used the colorization scheme in [5] (Figure 4(f)) to visualize these two flow fields in column (4) and (5).

Our human-annotated motion is significantly better than what the flow algorithm could achieve. Based on the criteria of the visual inspection of motion, the discontinuities of the annotated flow fields align well with the object boundaries, and the smoothness reflects the rigidity of the objects. In comparison, the flow fields computed by the flow algorithm often fail to capture the object boundary and the correct smoothness. The flow computation can have large errors for blurry and noisy regions, such as sequence (a) and (b) where the background motion is affected by the foreground in flow computation. In (c), (d), (f), (g) and (h), ambiguous boundary ownership causes the flow to mis-propagate across occluding boundaries. The mean AAE and AEP errors of each sequence are listed in the caption of Figure 6. Most of these errors are significantly larger than those for the Yosemite sequence. This suggests that our motion ground-truth database is more challenging for motion estimation algorithms and can be useful for developing better algorithms.

Now that we have sufficient realistic, ground-truth motion data, as a side effect, we can learn the statistics of realistic motion fields. These statistics can lead to more accurate prior of flow fields and help to improve flow estimation algorithms [17]. We computed the marginal and joint statistics of the ground-truth flow in our database and displayed the log histograms in Figure 5. In (a) and (b), the marginal of u (horizontal flow) is flatter than that of v (vertical flow), indicating that horizontal motion dominates vertical. As shown in (b) and (i), the marginal of v is asymmetric, and there are more pixels falling down than going up (due to



(a) (b) (c) (d) (e) (f) (g) (h)

(1) Frame (2) Layer labeling (3) Annotated motion (u) (4) Annotated motion (5) Optical flow estimation

Figure 6. Some frames of the ground-truth motion database we created. We obtained ground-truth flow fields that are consistent with object boundaries, as shown in column (3), the horizontal component of flow, and column (4), flow colorization using Figure 4 (f). In comparison, the output of an optical flow algorithm [8] is shown in column (5). The error between the ground-truth motion (4) and flow estimation (5) is as follows (AAE, AEP), (a): 8.996°, 0.976; (b): 58.904°, 4.181; (c): 2.573°, 0.456; (d): 5.313°, 0.346; (e) 1.924°, 0.085; (f): 5.689°, 0.196; (g): 5.2431°, 0.3853; and (h): 13.306°, 1.567. Most of the errors are significantly larger than the errors with the Yosemite sequence (AAE 1.723°, AEP0.071) . The parameter of the flow algorithm in column (5) is tuned to generate the best result for each sequence.

gravity). The marginals of the 1st-order derivatives of the flows are sparse, as shown in (c) to (f). Unlike the marginals of synthetic flow fields [17], our statistics show that the vertical flow is sparser than the horizontal flow, consistent with the fact that horizontal motion has a larger range. The temporal derivatives of the flow are not as sparse as the spatial ones, as depicted in (g) and (h). The joint histogram in (j) suggests that horizontal and vertical motion are likely to increase or decrease together temporally. The joint histograms in (k) and (l) reveal that the discontinuities of the flow are isotropic. At motion discontinuities, the change of vertical motion may dominate the change of horizontal motion, and vice versa, as shown in (m) and (n).

6. Conclusion

Motion analysis algorithms have been in use for decades, but little has been done to obtain the ground-truth motion for real-world videos. We presented a methodology and system to obtain ground-truth motion through human annotation. Our system is built upon several state-of-the-art motion analysis algorithms that allow the annotation of every pixel and every frame. A special graphical user interface is designed to allow the user to label layers, inspect motions, select parameters, and specify sparse correspondences. Our methodology is validated by comparison with the ground-truth motion obtained through other means and by measuring the consistency of human annotation. Using our system, we collected a motion ground-truth database consisting of challenging real-world videos for algorithm evaluation and benchmarking. We hope this database and our annotation code will lead to improved algorithms for optical flow and layered motion analysis.

Acknowledgement

Funding for this research was provided by NGA NEGI-1582-04-0004, Shell Research, and ONR-MURI grant N00014-06-1-0734.

References

[1] E. H. Adelson. Layered representations for image coding. Vision and Modeling Technical Report 181, MIT Media Laboratory, 1991.

[2] A. Agarwala, A. Hertzmann, D. H. Salesin, and S. M. Seitz. Keyframe-based tracking for rotoscoping and animation. *ACM SIGGRAPH*, 23(3):584–591, 2004.

[3] L. Alvarez, R. Deriche, T. Papadopoulo, and J. Sánchez. Symmetrical dense optical flow estimation with occlusions detection. In *Proc. ECCV*, pages 721–735, 2002.

[4] I. Austvoll. *A Study of the Yosemite Sequence Used as a Test Sequence for Estimation of Optical Flow*, pages 659–668. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2005.

[5] S. Baker, D. Scharstein, J. Lewis, S. Roth, M. J. Black, and R. Szeliski. A database and evaluation methodology for optical flow. In *Proc. ICCV*, 2007.

[6] M. J. Black and P. Anandan. The robust estimation of multiple motions: parametric and piecewise-smooth flow fields. *Computer Vision and Image Understanding*, 63(1):75–104, January 1996.

[7] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *Proc. ECCV*, pages 25–36, 2004.

[8] A. Bruhn, J. Weickert, , and C. Schnörr. Lucas/Kanade meets Horn/Schunk: combining local and global optical flow methods. *IJCV*, 61(3):211–231, 2005.

[9] Y.-Y. Chuang, A. Agarwala, B. C. Y.-Y. Chuang, A. Agarwala, B. Curless, D. H. Salesin, and R. Szeliski. Video matting of complex scenes. In *ACM SIGGRAPH*, 2002.

[10] B. K. P. Horn and B. G. Schunck. Determing optical flow. *Artificial Intelligence*, 17:185–203, 1981.

[11] M. Isard and A. Blake. Condensation – conditional density propagation for visual tracking. *IJCV*, 29(1):5–28, 1998.

[12] Y. Li, J. Sun, and H.-Y. Shum. Video object cut and paste. In *ACM SIGGRAPH*, 2005.

[13] C. Liu, W. T. Freeman, and E. H. Adelson. Analysis of contour motions. In *NIPS*, 2006.

[14] C. Liu, A. Torralba, W. T. Freeman, F. Durand, and E. H. Adelson. Motion magnification. In *ACM SIGGRAPH*, pages 519–526, 2005.

[15] B. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *Proc. of the Intl. Joint Conf. on Artificial Intelligence*, pages 674–679, 1981.

[16] D. Martin, C. Fowlkes, D. Tal, and J. Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proc. ICCV*, pages 416–423, 2001.

[17] S. Roth and M. Black. On the spatial statistics of optical flow. *IJCV*, 74(1):33–50, 2007.

[18] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: a database and web-based tool for image annotation. *IJCV*, 2008.

[19] P. Sand and S. J. Teller. Particle video: long-range motion estimation using point trajectories. In *Proc. CVPR*, volume 2, pages 2195–2202, 2006.

[20] J. Shi and C. Tomasi. Good features to track. In *Proc. CVPR*, pages 593–600, 1994.

[21] R. Szeliski. Fast surface interpolation using hierarchical basis functions. *TPAMI*, 12(6):513–528, 1990.

[22] R. Szeliski. Image alignment and stiching: A tutorial. *Foundations and Trends in Computer Graphics and Computer Vision*, 2(1), 2006.

[23] P. H. S. Torr, R. Szeliski, and P. Anandan. An integrated Bayesian approach to layer extraction from image sequences. *TPAMI*, 23(3):297–304, 2001.

[24] J. Wang, P. Bhat, A. Colburn, M. Agrawala, and M. Cohen. Interactive video cutout. In *ACM SIGGRAPH*, 2005.

[25] J. Y. A. Wang and E. H. Adelson. Representing moving images with layers. *IEEE Trans. Image Processing*, 3(5):625–638, 1994.

[26] Y. Weiss and E. H. Adelson. Perceptually organized EM: A framework for motion segmentation that combines information about form and motion. Technical Report 315, MIT Media Lab Perceptual Computing Section, 1994.

[27] J. Wills, S. Agarwal, and S. Belongie. What went where. In *Proc. CVPR*, pages 37–44, 2003.