Flowdometry: An Optical Flow and Deep Learning Based Approach to Visual Odometry

Peter Muller
Rochester Institute of Technology
pmm5983@rit.edu

Andreas Savakis
Rochester Institute of Technology
andreas.savakis@rit.edu

Abstract

Visual odometry is a challenging task related to simultaneous localization and mapping that aims to generate a map traveled from a visual data stream. Based on one or two cameras, motion is estimated from features and pixel differences between frames. Because of the frame rate of the cameras, there are generally small, incremental changes between subsequent frames where optical flow can be assumed to be proportional to the physical distance moved by an egocentric reference, such as a camera on a vehicle. In this paper, a visual odometry system called Flowdometry is proposed based on optical flow and deep learning. Optical flow images are used as input to a convolutional neural network, which calculates a rotation and displacement for each image pixel. The displacements and rotations are applied incrementally to construct a map of where the camera has traveled. The proposed system is trained and tested on the KITTI visual odometry dataset, and accuracy is measured by the difference in distances between ground truth and predicted driving trajectories. Different convolutional neural network architecture configurations are tested for accuracy, and then results are compared to other state-of-the-art monocular odometry systems using the same dataset. The average translation error from the Flowdometry system is 10.77% and the average rotation error is 0.0623 degrees per meter. The total execution time of the system per optical flow frame is 0.633 seconds, which offers a 23.796x speedup over state-of-the-art methods using deep learning.

1. Introduction

Visual odometry (VO) is a challenging task that generates a map of where a vehicle has traveled given an input video stream from camera(s) mounted on the vehicle. This type of VO capability can assist in the development of autonomous vehicles and robots and create maps of various types of spaces, among many other applications. There is a need for autonomous vehicles and robots to be able to track where they have been, because this knowledge helps to create predictive models for route understanding and navigation. An important step in accumulating the mapping over the entire length of the video sequence is estimating the movements strictly between two consecutive frames in the sequence. This paper proposes Flowdometry, a system for inter-frame odometry estimation based on optical flow and deep learning.

In traditional simultaneous localization and mapping (SLAM) approaches, the final results are refined over the course of the run-time of the algorithms. Particle filtering [7], bundle adjustment [9], and feature tracking [10] have all shown highly accurate results in this field. Each is capable of relocalizing within the generated maps, which allows for them to correct scale and rotation drifts from previously visited places in the map. However, this is not the case for systems that use inter-frame estimation only. When only considering deltas from one frame to the next, error is accumulated over the length of the driving path, which means that early error will propagate all the way to the end of the video sequence in addition to any other error along the way. Some examples of these types of systems are VISO2-M [6], SVR VO [2], and P-CNN [3], which will all be covered in more detail in Section 2.

One way to estimate motion between two images or frames in a video sequence is to relate physical movement to the optical flow between the images. Optical flow is a method of densely computing how far a pixel moves from one image to the next. There are several highly accurate algorithms for this calculation, such as the Brox algorithm [1], DeepMatching [11], EpicFlow [12], and DeepFlow [13].

One major issue with the optical flow algorithms is that they take a significant amount of time per image pair to run. Thus, computational efficiency is an important consideration, especially for vehicular applications where there is an ever-increasing need for fast computation.

FlowNet [4] was a deep learning approach to calculating optical flow between images. Due to the use of graphics processing units (GPUs), the run time of a calculation

978-1-5090-4822-9/17 $31.00 © 2017 IEEE
DOI 10.1109/WACV.2017.75

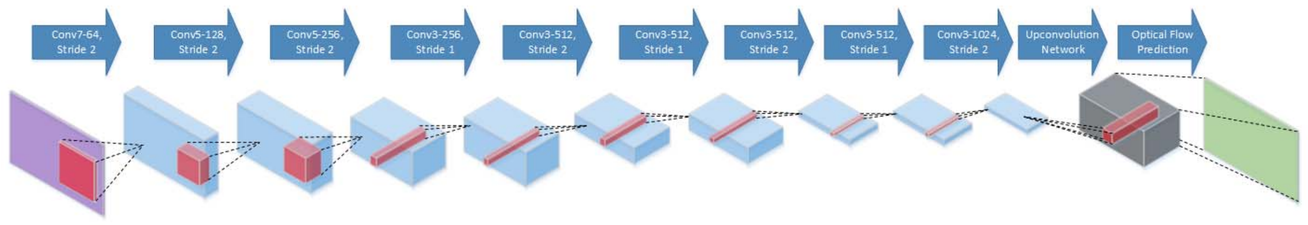


Figure 1: FlowNetS architecture with the contractive side of the network shown. The upconvolutional stage takes intermediate results from previous convolutions to assist in ultimately forming a dense optical flow output image.

was reduced to fractions of a second, making it the fastest running optical flow system, but at the cost of not being as accurate as the state-of-the-art methods. Two different network architectures, FlowNetC and FlowNetS, were introduced, and it was shown that FlowNetC generalized better to synthetic scenes while FlowNetS generalized better to naturalistic scenes.

This paper contributes an end-to-end deep learning solution for visual odometry inspired by FlowNetS. The proposed system utilizes a novel network input of raw optical flow in a repurposed neural network architecture that does not require semi-supervised pretraining. Due to implementation choices, this system has a greatly reduced execution time from raw camera data to odometry results compared to any other similar system.

The rest of this paper is organized as follows. Section 2 describes related works, Section 3 details the Flowdometry architecture, Section 4 presents the results, and Section 5 summarizes the work.

2. Related Works

The visual odometry task differs from other SLAM works in that strictly inter-frame odometry is calculated and accumulated over the sequence, with no other methods of error correction or post processing to relocalize the vehicle within the map. These systems are subject to integration error from the accumulation of small errors over the length of the video sequence being processed.

The FlowNetS architecture [4] was shown to generalize natural scenes well when computing optical flow. The contractive architecture of FlowNetS is shown in Fig. 1. Using ten layers of convolutions with and without striding, two input images were processed down into an activation volume that was then upconvolved into the optical flow estimation. The upconvolution was defined as a convolution in which the output activation volume had greater height and width dimensions than the input volume. To assist in the upconvolutions, activations from the contractive convolutions were appended to the upconvolution inputs and processed at the same time. Intermediate flow estimations were also used to increase the amount of data available for each upconvolution. All convolutions used a leaky rectified linear unit (leaky ReLU) activation function. The output of the convolutions is then upscaled using linear interpolation to the full resolution of the input images to create the final optical flow prediction.

In order to provide a sufficient amount of training data for such a large network, the authors of FlowNet devised the Flying Chairs dataset, which included over 20,000 image pairs of synthetic images to train on. All example pairs included a chair overlaid on a random background, with the second image being a transformation of the first so that optical flow could be computed between the two images.

In other works on VO, a frame-to-frame movement estimation system was developed in [6], which algorithmically calculated a three dimensional disparity between frames, in a system called VISO2-M. The original intent of this work was to generate 3D maps in real-time based on stereo video sequences. This is done by computing the optical flow between image pairs and relating the dense results to depth measurements. Movement for a SLAM algorithm, as adapted in [3], could be derived by corresponding the disparities to the camera sources over the sequence of video.

In [2], a support vector machine (SVM) approach, called SVR VO, was taken so that the system could ”learn” how to relate optical flow calculation to physical movement of the camera source. This was one of the first known methods that used a non-geometric approach to visual odometry. It was shown that with labeled data, learned approaches, such as SVM or Gaussian process regression, can perform as well or better than the geometric systems before them. In addition, the learned systems ran much more quickly and efficiently than the geometric ones.

In [8], a convolutional neural network model was used for producing odometry information between frame pairs in video sequences. Unable to regress accurate values with that method, the problem was converted into a classification approach by discretizing the ground truth data into ranges of rotations and displacements. This could be due to the fact that there were very few parameters to train in the network architecture that was used. It only contained two convolu-

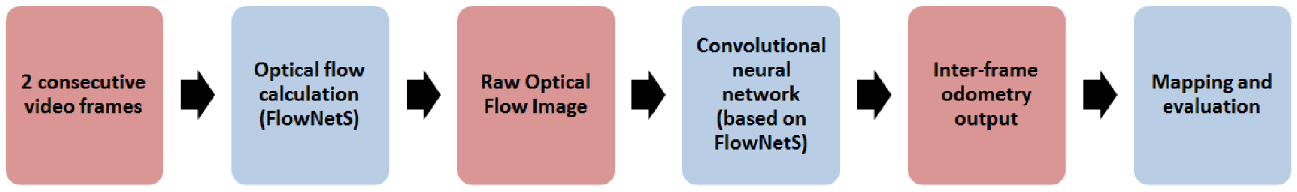

2 consecutive video frames
Optical flow calculation (FlowNetS)
Raw Optical Flow Image
Convolutional neural network (based on FlowNetS)
Inter-frame odometry output
Mapping and evaluation

Figure 2: The Flowdometry system overview. Raw camera frames are input into the FlowNetS network to produce raw optical flow images. These raw flow images are used as input for the odometry network, which is based on the FlowNetS architecture. The odometry results are then used for mapping and evaluation of the system.

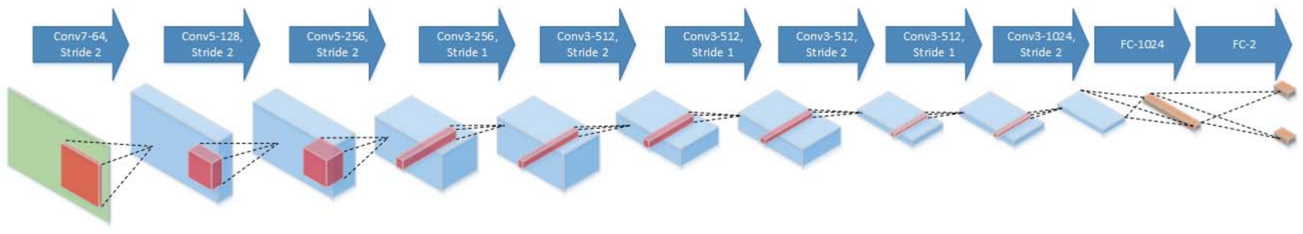


Figure 3: The Flowdometry convolutional neural network architecture based on the contractive part of FlowNetS. The architectures are the same except for the input data and the output past the last convolution in the contractive network.

tional layers and a fully-connected layer, so it did not have the learning capacity that a deeper network would have.

Later, in [3], a convolutional neural network that could regress these values was introduced, and this surpassed the accuracy of other inter-frame odometry estimation systems. The P-CNN VO system, as the authors called it, was a combination of five convolutional neural networks, one for each quadrant of an image and one for the image as a whole, working together to produce translation and rotation data. Although the individual networks only had two convolutional layers, the best results were achieved when combining the five networks together, meaning that the number of parameters is, at the minimum, five times greater than that of a single network with two convolutional layers. This increases the learning capacity of the model, which results in the potential for more accurate results.

3. Flowdometry Architecture

The proposed system calculates odometry estimation in three steps. First, raw optical flow is computed from the FlowNetS architecture. Next, the flow image is used as input to the repurposed FlowNetS network, which will henceforth be referred to as Flowdometry, to produce inter-frame odometry estimations. In a final post-processing step, the estimations are accumulated to generate maps and accuracy measurements. This system is represented in Fig. 2.

3.1. Network Architecture

The convolutional neural network used to regress odometry values is based on the contractive half of the FlowNetS [4] network. Unlike other works, Flowdometry uses raw optical flow, a two-channel image that describes horizontal and vertical pixel movement, as input. The other works used color images that were converted from the raw optical flow to color representations of the optical flow vectors. FlowNetS is used to generate optical flow images for each pair of consecutive video frames. The resulting optical flow image is used as input to the Flowdometry architecture. FlowNetS was chosen to produce the optical flow input because it was shown to generalize better to natural scenes and also because it ran much faster than other optical flow algorithms.

For the regression part of the system, a modified FlowNetS architecture was chosen. Based on Fig. 1, the changes were that the input was now a two-channel, instead of six-channel image, and the refinement network was replaced by a fully-connected layer followed by the output nodes for regressing rotation and displacement information between frames. It was hypothesized that the modified architecture could produce the physical motion results because the FlowNet network was capable of capturing motion information. The regression network structure contributed in this work is shown in Fig. 3.

To train this network, the ground truth data from the KITTI odometry benchmark were transformed into incremental movement values of an angle and a displacement. A raw optical flow image, representing two consecutive frames in the video sequences, was associated with the incremental odometry values, and then the network was trained with over 114,000 training samples, including the mirrored video sequences. Unlike P-CNN, this network did not require semi-supervised weight pretraining with an auto-encoder network before training it to regress the final values. This speeds up the training time, and simplifies the training process as a whole.

For each consecutive frame pair, the incremental movements are calculated by the neural network. The rotations and displacements are accumulated over the length of the video sequences, and the results are mapped and reformatted into the rotation matrix style that the KITTI odometry evaluation requires. Because each measurement is relative to the current orientation of the vehicle, the movements are projected onto a two dimensional plane, and the relative rotation is applied, followed by the displacement in that direction. In this application, there is an assumption of linearity for each optical flow image because it is not possible for the vehicle to move too far in the span of the 10Hz refresh rate of the camera.

4. Results

4.1. Dataset

The KITTI odometry benchmark [5] is a dataset that provides color or grayscale video from mounted stereo cameras on a vehicle. The video contains driving scenes in suburban and highway environments. Corresponding to each frame of the video is a ground truth transformation matrix that transforms the current frame into the coordinate system of the first frame.

Assuming that the vehicle starts at the origin of a map, the rotation matrix translates and rotates the vehicle in the current frame, relative to the initial starting point. The evaluation generally gives 11 sequences with ground truth data to tune an odometry system and 11 sequences without ground truth to use for online evaluation of the system. By nature of the integration error introduced in inter-frame odometry applications, these systems are not evaluated online against the state-of-the-art SLAM systems. Instead, these systems are trained or tuned on the first 8 video sequences with ground truth and evaluated on the remaining three sequences with ground truth, ensuring that those results were not used as part of the training process. Not all of the 11 sequences have the same number of video frames. Table 1 shows the number of frames in each video sequence.

The video sequences in this dataset do not contain nearly as many examples of the vehicle turning as it going straight, which makes regressing accurate odometry values difficult in these situations. To provide more data in this work, the entire dataset was mirrored and ground truth data were adjusted in addition to the original data to give the network twice as much data to train with.

Phase	Sequence	Frames
Training	00	4541
	01	1101
	02	4661
	03	801
	04	271
	05	2761
	06	1101
	07	1101
Testing	08	4071
	09	1591
	10	1201

Table 1: Number of frames per video sequence in the KITTI odometry benchmark. Adding mirrored sequences doubles the amount of training data available to the system.

4.2. Evaluation

For evaluation of the Flowdometry application, results are compared to the results of the regression models introduced in [2, 3, 6]. It should be noted that Flowdometry was developed independently of these works.

Using the KITTI odometry evaluation software kit, the maps were plotted against the ground truth, and average errors for each sequence and all sequences combined were calculated. Figs. 4, 5, and 6 show the driving paths that the application generated from the optical flow input and incremental odometry output. Due to the integration error accumulated over the sequence lengths, the resulting paths tend to look ”stretched” away from the ground truth values, but based on the shape and scale of the estimated path, it shows that many of the small, incremental estimates were good approximations. The largest source of error was around turns, where the rotation estimate was either too large or too small to accurately navigate those areas.

Average error results for Flowdometry, as well as comparisons to VISO2-M, SVR VO, and P-CNN, are presented in Table 2. These results show that Flowdometry surpassed the translational accuracy of the non-deep learning approaches of VISO2-M and SVR VO, and it was comparable to the state-of-the-art deep learning model of P-CNN. This system produced more consistent error across each video sequence, and exceeded the state-of-the-art model in Sequence 10. The rotational error was higher for each sequence in Flowdometry because only one axis of rotation was considered, even though the vehicle was subjected to all three axes of rotation in reality. Our system is less complex than P-CNN in terms of the training method and number

	VISO2-M [6]		SVR VO [2]		P-CNN VO [3]		Flowdometry	
Seq	Trans [%]	Rot [deg/m]	Trans [%]	Rot [deg/m]	Trans [%]	Rot [deg/m]	Trans [%]	Rot [deg/m]
08	19.39	0.0393	14.44	0.0300	**7.60**	**0.0187**	9.98	0.0544
09	9.26	0.0279	8.70	0.0266	**6.75**	**0.0252**	12.64	0.0804
10	27.55	0.0409	18.81	**0.0265**	21.23	0.0405	**11.65**	0.0728
Avg.	18.55	0.0376	13.81	0.0302	**8.96**	**0.0235**	10.77	0.0623

Table 2: Average translation and rotation errors for each video sequence and method of visual odometry. Results for the works in comparison are provided from the P-CNN VO [3] work.

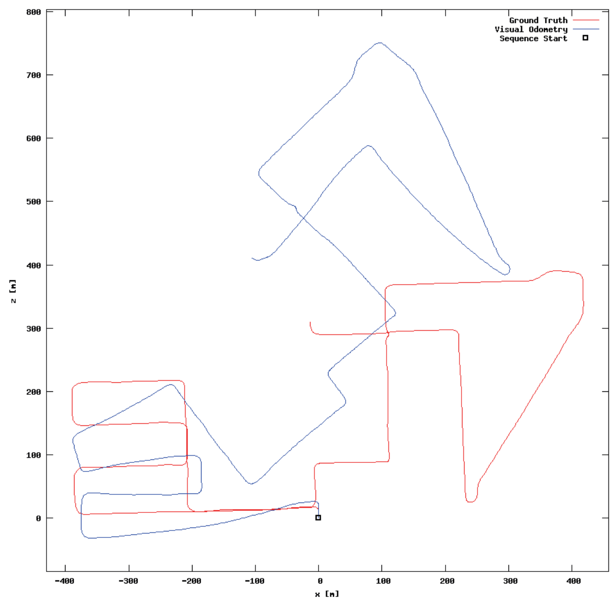


Figure 4: Sequence 08 odometry results compared to the ground truth path.

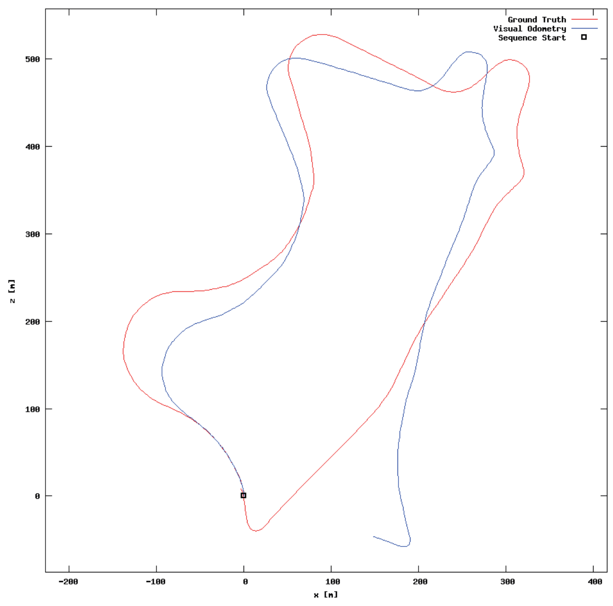


Figure 5: Sequence 09 odometry results compared to the ground truth path.

of different networks to concatenate together for regressing the final results.

The odometry benchmark calculated rotation and translation errors as functions of sequence length and vehicle speed. Fig. 7 shows graphs of each combination of error and cause of error for the average of all test sequences. When measuring against the video sequence length, the error tends to decrease as the length of the video subsequence increases. This shows that there may be large spikes in error at certain points in the sequence, but they get averaged to smaller errors over a longer period of time. This is especially evident in Fig. 7a. Contrasting those charts, when the errors are compared to the vehicle speed, there are larger errors when the vehicle is traveling at its fastest. This is most likely due to the fact that there is minimal data in each sequence where the vehicle is traveling that quickly. With fewer samples of fast driving, there is more significance to the amount of error that is present when the car does achieve those speeds.

Due to the different number of frames in each of the testing video sequences, any results aggregated over sequence length will be biased, especially toward Sequence 08 which has many more frames than the other two sequences combined. In contrast, the smaller sequences had more variation, however brief, in car speed, which led to the spikes seen in Figs. 7b and 7d. Because of the limited data in these buckets, the results are weighted to the available data, which in these cases, correspond to high error at these speeds. With few data points to average with, the highest error contributes most to the shape of the error chart.

To show that the Flowdometry system was trained to generalize to the given results, an example plot from Se-

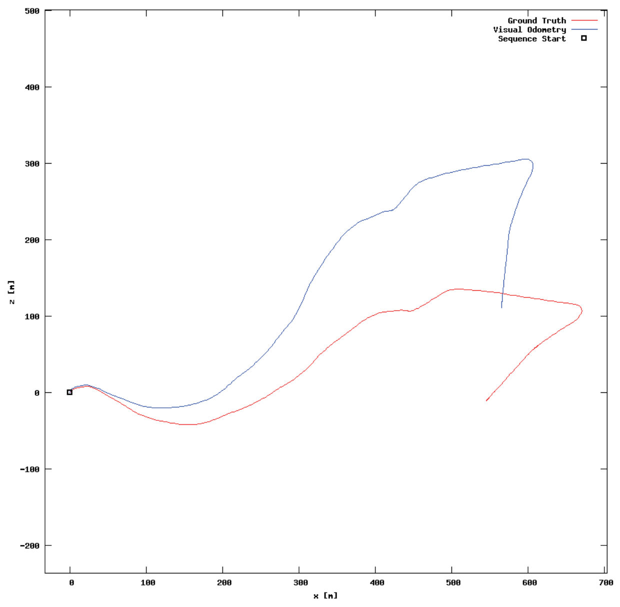
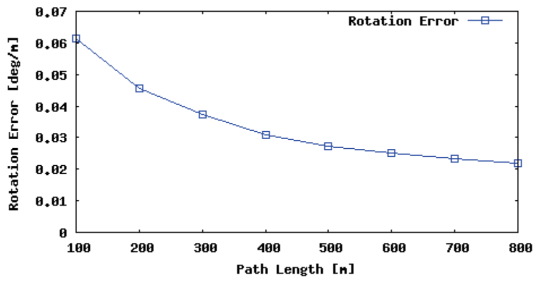


Figure 6: Sequence 10 odometry results compared to the ground truth path.

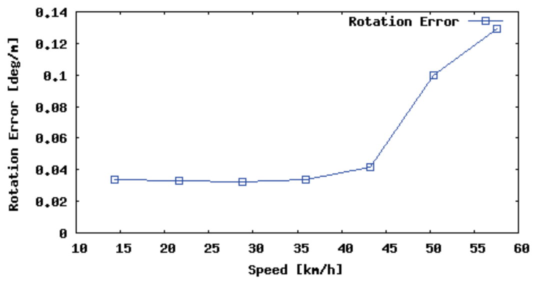
quence 00, one of the training sequences, is shown in Fig. 8. Just as in the testing sequence plots, integration error propagates throughout the length of the sequence, but distinct shapes of the driving path can be seen from the VO system. The model does not overfit to the training data because even in Sequence 00, turns were difficult for the system to accommodate, especially the long, gradual left turn at the end of the sequence. In this case, if more recordings of driving paths were taken with the same camera setup, it would be expected that this system provides results consistent with what has been measured with Sequences 08, 09, and 10. Although the model did not overfit the data, it is likely that there was not enough training data to create the most accurate model. As visualized in the driving paths, the scale of movement is accurate, but the turning angles are the largest source of error. This suggests that the Flowdometry model learned how to accurately determine the distance traveled between video frames, but it was not able to learn the magnitude of a turning angle, most likely because there are not many examples of turning in the training dataset.

4.3. Timing

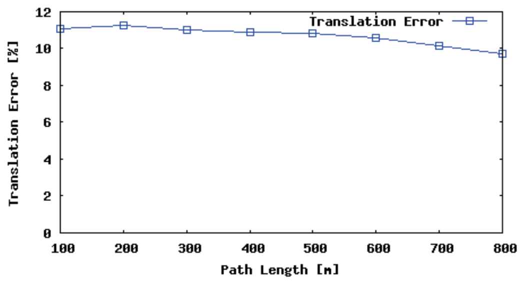
One of the advantages of the Flowdometry system is its overall computation time from input images to odometry output. Because it uses raw optical flow as input to the odometry neural network, it skips a timing penalty associated with converting optical flow to RGB images as done in the compared works. In addition, using FlowNetS instead



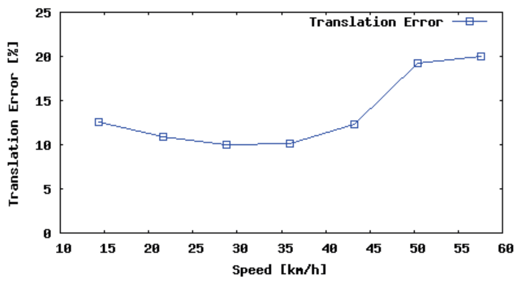
(a) Average rotation error vs. video sequence length.



(b) Average rotation error vs. vehicle speed.



(c) Average translation error vs. video sequence length.



(d) Average translation error vs. vehicle speed.

Figure 7: Average errors across all sequences.

of the Brox algorithm greatly increases the rate at which input images are converted to optical flow, but this comes

	Optical Flow Calculation [s/frame]	RGB Conversion [s/frame]	Odometry Calculation [s/frame]	Total Execution [s/frame]	Flowdometry Speedup
VISO2-M [6]	14.812	0.210	0.063	15.085	23.831
SVR-VO [2]	14.812	0.210	0.112	15.134	23.908
P-CNN [3]	14.812	0.210	**0.041**	15.063	23.796
Flowdometry	**0.271**	**0.000**	0.362	**0.633**	—

Table 3: Timing information for each system. The speedup column expresses how much faster the total execution time of Flowdometry is in comparison to the total execution time of each system.

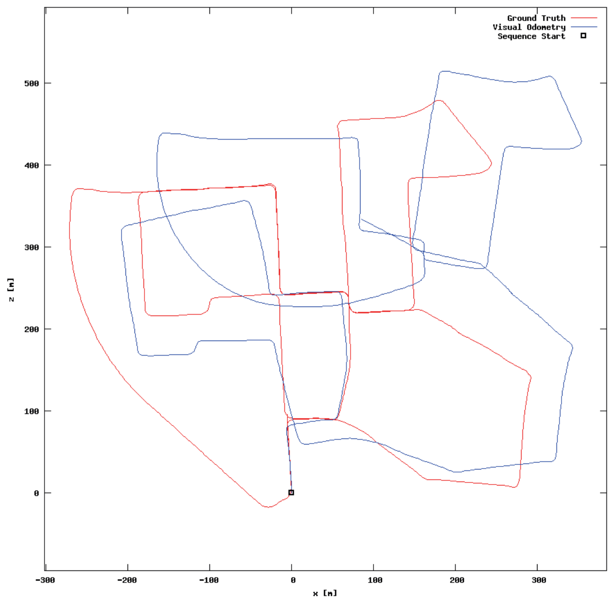


Figure 8: Sequence 00 odometry results compared to the ground truth path. This sequence was one of the training sequences for the Flowdometry system, and it shows that the system did not overfit to training data.

with the cost of having less accurate optical flow images. Table 3 shows the comparison of execution times between each of the systems. Despite the longer execution time of the odometry calculation for Flowdometry, using FlowNetS and raw optical flow input greatly reduces the total amount of time required to produce odometry results. The Brox algorithm was used to calculate optical flow for images in the KITTI dataset and the lowest observed time for 1000 different image pairs was recorded for the comparison systems. For measuring the FlowNetS optical flow calculation for the Flowdometry result, the average execution time for 1000 image pairs was measured. Conversion to RGB images from optical flow was measured as the average execution time over 1000 different optical flow images. Flowdometry skipped this step so its measured time is 0.00 seconds. The odometry timings for the comparison systems are the best recorded values provided in [3]. For Flowdometry, the average execution time over 1000 input optical flow images was measured. It is found that Flowdometry has 23.796x speedup over the next fastest system. All measurements that were not provided by Costante et al. were calculated on an Ubuntu 15.10 x86_64 desktop computer with a 3.00GHz Intel Core 2 Duo CPU and NVIDIA GeForce GTX 950 GPU.

5. Conclusions

The Flowdometry application introduces several contributions. An end-to-end convolutional neural network system takes two consecutive video frames as input, converts them to an optical flow image, then regresses odometry information based on the raw flow data. The architecture repurposes the FlowNetS architecture to perform the odometry regression. Semi-supervised pretraining was not required to achieve results on par with the state-of-the-art inter-frame odometry systems. Because of the use of neural networks for all aspects of the system, Flowdometry is also the fastest running odometry system of its kind.

The results of this system show that it is comparable with other systems of similar application. Its rotation and translation accuracies surpassed those of non-deep learning approaches, and were similar to the results of the deep learning approach. As in other systems of inter-frame odometry estimation, integration error plays a large role in the final shape of the map generated from the neural network output data. Although this system tended to follow simpler practices than the state of the art model, it was still comparable in translation accuracy, even surpassing the state of the art in one of the video sequences. Our system uses raw optical flow input instead of a color image to produce odometry data that is comparable to the state of the art. This is the first system, as of our knowledge, to use neural networks all the way from generating the optical flow information to producing the odometry output.

There are several opportunities for future improvements in follow up work. The CNN architecture may be reduced in size so that similar results may be achieved with fewer parameters and lower network training time. More training data along with dataset augmentation could help to more accurately regress turns. As it currently stands, there are many variations of turns with few examples of each, making it difficult for the system to accurately regress values for them. It is worth exploring other types of network architectures, that may be able to more accurately regress turns, particularly as more frame data become available for training.

References

[1] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High Accuracy Optical Flow Estimation Based on a Theory for Warping. In *Computer Vision - ECCV 2004*, volume 3024, pages 25–36. Springer Berlin Heidelberg, 2004.

[2] T. A. Ciarfuglia, G. Costante, P. Valigi, and E. Ricci. Evaluation of non-geometric methods for visual odometry. *Robotics and Autonomous Systems*, 62(12):1717–1730, dec 2014.

[3] G. Costante, M. Mancini, P. Valigi, and T. A. Ciarfuglia. Exploring Representation Learning With CNNs for Frame-to-Frame Ego-Motion Estimation. *IEEE Robotics and Automation Letters*, 1(1):18–25, jan 2016.

[4] P. Fischer, A. Dosovitskiy, E. Ilg, ..., and T. Brox. FlowNet: Learning Optical Flow with Convolutional Networks. *Iccv*, page 8, 2015.

[5] A. Geiger, P. Lenz, and R. Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 3354–3361, 2012.

[6] A. Geiger, J. Ziegler, and C. Stiller. StereoScan: Dense 3d reconstruction in real-time. In *IEEE Intelligent Vehicles Symposium, Proceedings*, pages 963–968. IEEE, jun 2011.

[7] G. Grisetti, C. Stachniss, and W. Burgard. Improved techniques for grid mapping with Rao-Blackwellized particle filters. *IEEE Transactions on Robotics*, 23(1):34–46, 2007.

[8] K. Konda and R. Memisevic. Learning Visual Odometry with a Convolutional Network. In *Proc. International Conference on Computer Vision Theory and Applications*, 2015.

[9] K. Konolige and M. Agrawal. FrameSLAM: From bundle adjustment to real-time visual mapping. *IEEE Transactions on Robotics*, 24(5):1066–1077, 2008.

[10] R. Mur-Artal, J. M. M. Montiel, and J. D. Tardos. ORB-SLAM: A Versatile and Accurate Monocular SLAM System. *IEEE Transactions on Robotics*, 31(5):1147–1163, 2015.

[11] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. DeepMatching: Hierarchical Deformable Dense Matching. *International Journal of Computer Vision*, (Lowe 2004):1–24, 2016.

[12] J. Revaud, P. Weinzaepfel, Z. Harchaoui, C. Schmid, J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. S. E. Edge. EpicFlow : Edge-Preserving Interpolation of Correspondences for Optical Flow. *Cvpr 2015*, 2015.

[13] P. Weinzaepfel, Z. Harchaoui, C. Schmid, P. Weinzaepfel, Z. Harchaoui, C. Schmid, Z. Harchaoui, and C. Schmid. DeepFlow : Large displacement optical flow with deep matching. pages 1385–1392, 2013.