

Using learning of speed to stabilize scale in monocular localization and mapping

Duncan P Frost David W Murray Victor A Prisacariu
Active Vision Laboratory, Department of Engineering Science
University of Oxford, OX1 3PJ, UK.

[duncan, dwm, victor]@robots.ox.ac.uk

Abstract

Monocular visual localization and mapping algorithms are able to estimate the environment only up to scale, a degree of freedom which leads to scale drift, difficulty closing loops, and eventual failure. This paper describes an image-driven approach for scale-drift correction which uses a convolutional neural network to infer the speed of the camera from successive monocular video frames. We obtain continuous drift correction, avoiding the need for explicit higher-level representations of the map to resolve scale. We also propose a novel method of including speed estimates as a regularizer in bundle adjustment which avoids the pitfalls of sudden imposition of scale knowledge. We demonstrate our approach using long-distance sequences for which ground truth is available, and find output that is essentially free of scale drift. We compare the performance with number of other methods for scale-drift correction from monocular data, and show that our solution achieves more accurate results.

1. Introduction

The last three or so decades have seen remarkable developments in all categories and varieties of visual 3D scene reconstruction — monocular and multicamera, static and dynamic, sparse and dense, calibrated and not, and so on. In recent years there have been similarly transformational advances in forging more direct links between images and scenes through learning, using both traditional techniques and, most recently, deep approaches. The grand challenge in camera localization and scene recovery is now no longer one of geometry *per se*, but is one of combining these approaches — one of determining when and how best to use learnt processes and representations not only to improve sections of the geometry processing, but also to make the geometry something more than the sum of its points.

In this paper we apply learning at a low level, enhanc-

ing the geometry pipeline in monocular simultaneous localization and mapping by first demonstrating that, at least for predominantly divergent image flow fields, learning permits the direct recovery of per-frame camera speed from successive pairs of images in a sequence. A second and equally important contribution is the development of a regularization mechanism to allow those speeds to tame scale drift resulting from the speed/scaling ambiguity. We show in experiment that the combined method provides better *stabilization* of scale than (i) using the speed directly as odometry, (ii) using a constraint on camera height above a ground plane [13] or (iii) using recognized objects of known size [11]. Fig. 1 shows recovered camera trajectories compared with ground truth estimated first (a) without and then (b) with the recovered per-frame speeds and speed-regularized bundle adjustment.

All simultaneous localization and mapping algorithms that use sensors which deliver relative measurements (*i.e.* that use image measurements derived from relative or differential movement) are predisposed to accumulate drift over time. One can ascribe this to the inherent gauge freedoms in the maps that they build [32], and model it by in-

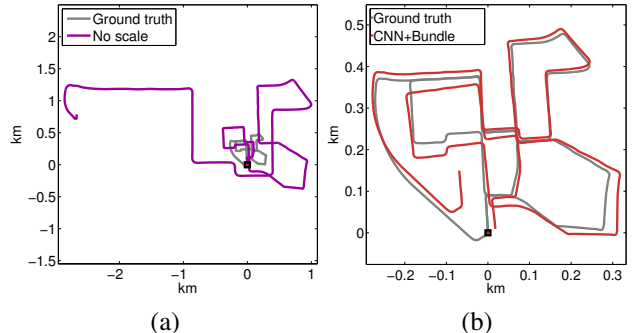


Figure 1. (a) Map produced by our monocular system without any scale stabilisation on KITTI sequence 00 and (b) a map produced by same system with our proposed method for scale stabilisation. The ground-truth trajectory for the sequence is shown for reference in both figures.

tegrating stochastic error in the state estimates. Methods using stereo, LiDAR, or RGBD input have six degrees of gauge freedom and thus drift in their rotational and translational estimates, but monocular localization and mapping has an additional degree of freedom as the estimated map is determined only up to scale. This results in additional drift where the scale of the map (or, equivalently, speed of the camera) becomes ever more erroneous.

Unlike translational or rotational drift, progressive scale-drift can rapidly incapacitate a monocular method. For example, in keyframe-based algorithms (e.g. [20]) the decision to add new keyframes is often based on the cameras proximity to previous ones. If the estimate of speed is too large, keyframes will be created too rapidly, and if too small they are not created at all. While loop-closure can redistribute and reduce accumulated error [14] [7], [28], [6], the error in location can be so severe, as evident in Fig. 1, that searching for closure is futile.

Scale drift in monocular reconstruction has been managed either by resorting to additional sensors or making assumptions about the geometry of camera or the environment or both. In the first category, vision is an indispensable partner to LiDAR, GPS, radar, and wheel odometry in all autonomous vehicle sensing packages. Somewhat self-defeating for monocular vision is the addition of second camera (e.g. [30], [9]), which immediately resolves scale if the baseline between the two cameras is known. A favoured non-visual sensor is the inertial measurement unit, able to measure acceleration and 3D orientation using a mix of accelerometers and gyroscopes. Often an EKF is used to fuse data [16, 24], although spline-fitting has been employed for a keyframe-based monocular method [18]. Other more exotic distance-sensitive sensors can be useful: for example, [1] uses an air pressure transducer to measure height changes in a micro air vehicle.

The immediate disadvantages of adding further sensors are the increases in both bulk and power consumption. Instead, assumptions about the structure of the scene have been used to infer information about scale. Long term monocular odometry methods often assume a fixed camera height above the ground-plane [13] [19] [27], [31]. Detecting the plane and enforcing the prior knowledge of the camera height allows the scale of the estimated local map to be corrected. Repetitive structure in the mapped environment has been also been leveraged to propagate scale-knowledge from areas with more certain scale to uncertain ones [4]. If an environment contains known objects with a consistent size, these may be incorporated into the map to provide local information about metric scale. Early work using objects in monoSLAM was described in [5], but the method of inclusion of size data into its extended Kalman filter was disruptive. More recently [11] showed that recognized object size could provide long-term scale stability in bundle

adjustment.

Here we wish to avoid enforcing additional constraints on the geometrical setup of the camera or the environment, and also to avoid the cost of additional sensors. Instead we train a convolutional neural network using pairs of monocular images with known camera displacement between capture. In use, the network takes a pair of consecutive images and returns the per-frame speed of the camera. By using all the image data, rather than some portion of it, it can operate in conditions where methods that use a portion of it (such as the ground-plane) cannot function due to occlusion. The only implicit requirement is a degree of similarity between the image motion between the images supplied to the network for training and deployment.

The paper is organized as follows. Section 2 details our network architecture and Section 3 introduces our method for incorporating network speed estimates into monocular localization and mapping using a novel bundle adjustment. Section 4 presents quantitative results for the method. Concluding remarks are offered in Section 5.

2. Direct speed inference from imagery

One method of determining differential camera motion might be first to recover depth for each image in a sequence. Recovering single-view depth was well-studied using traditional learning methods [23], [25], [26]. Unsurprisingly, while all accurately recover relative depths within an image, they struggle more to estimate absolute depths. More recent works have increased accuracy by using CNNs trained end to end [8, 22]. Eigen *et al.* [8] shows promising results for both absolute and relative depth estimates by using a pair of deep networks in a coarse to fine manner: an initial network globally predicts the entire depth map for a single image on a coarse level, the result of which is locally refined using a fine-scale network. However in practice using single-view depth maps for motion appears deeply flawed. Among several problems the most onerous is that accurate depth maps are required for training. Next, in use, depth matching would be needed per frame, to be followed by a solution of the absolute orientation problem.

Keeping the representation of pixel-by-pixel depth implicit and training using a simple scalar are the two key advantages of using more images. Turning to image pairs, Fischer *et al.* [10] use deep learning to estimate optical flow between two images. They compare two different network architectures, a convolutional network using a two-channel input, and a novel Siamese-style network that attempts to determine the correlation between features for each “wing” of the network. Slightly better results are obtained using the Siamese network. Konda and Memisevic [21] applied deep learning to visual odometry, but used several more images. Their method processes the five most recent frames from both the left and right stereo images of the KITTI dataset.

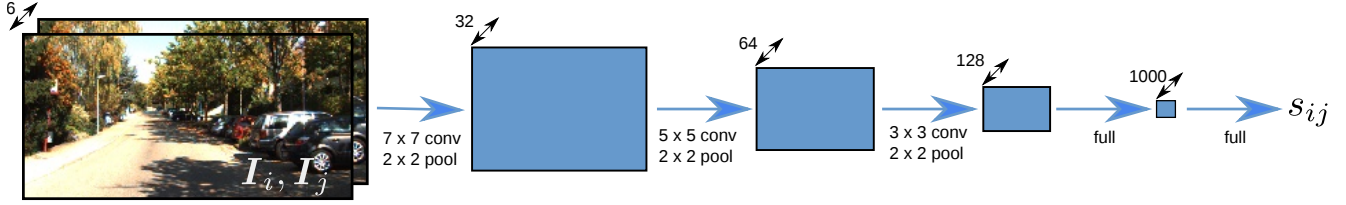


Figure 2. Network architecture used for speed regression from image pairs.

These are each combined in a Siamese-style network that predicts discretized camera motion, effectively reducing the problem to a classification task. The authors note that the amount of data in the KITTI dataset proved insufficient for full regression. Ego-motion is similarly used in [2, 17] to pre-train networks used for other tasks.

The work of Zagoruyko and Komodakis [36] suggests that using more than two images is unnecessarily cumbersome. They explore a number of architectures for the estimation of 2D similarity transformations between two images, including a Siamese network (which performs separate feature extraction operations on each image before combining the results), a pseudo-Siamese network (which uses shared weights in each feature-extraction stage), and a two-channel network (which combines both input images into a single two-channel image). The two-channel network is found to have superior performance.

More recently Zhou *et al.* [37] present an unsupervised method focused on joint estimation of ego-motion and depth from monocular video. The inferred transformations between frames are still, however, only accurate up to scale. Estimated transformations must therefore still be scaled up by their ground-truth values during ego-motion evaluation.

2.1. Network

The CNN developed here is inspired both by the global coarse-scale network in Eigen *et al.* [8] and the two channel structure of Zagoruyko and Komodakis [36]. In [8], relatively large initial convolutional layers are used to integrate a global understanding of the whole image. Similarly, we choose particularly large convolutions to increase the spatio-temporal receptive field of our network. For fast camera motion, the network needs to be able to account for large displacements between images.

For training, and in use, the input to our convolutional neural network is a 6-channel image consisting of the two RGB images I_i and I_j stacked together. This is fed into three convolutional layers where each convolution is followed by a 2×2 max-pooling layer. These are followed by two fully connected layers. The hyperbolic tangent function, $\tanh(\cdot)$, is used as a non-linearity between layers. Early experimentation showed that the hyperbolic tangent was better than the popular ReLU non-linearity at estimat-

ing *absolute* speed. The training loss is the Euclidean

$$E(s_{ij}, \hat{s}_{ij}) = (s_{ij} - \hat{s}_{ij})^2, \quad (1)$$

where \hat{s}_{ij} is the current output of the network, and s_{ij} is the desired output.

When training, the desired output is found from ground truth poses T_i and T_j . The relative pose between them is computed as $T_{ij} = T_j T_i^{-1}$ and the camera speed is found from the translational component in the rightmost column of T_{ij} as $s_{ij} = |t_{ij}|$. While the proposed network is relatively shallow in comparison with modern networks [35], this is for the purpose of maintaining real-time feed-forward performance. Moreover, the size of the network is more than sufficient for its given task, making more modern networks ultimately unnecessary.

2.2. Dataset and training

We use different sequences in the KITTI outdoor dataset for training and evaluation [12, 19]. We note from the out-set that although this dataset is compatible with the fixed camera-height assumption, this constraint is not used in our work, but will be used in experimental comparison later. The first eleven sequences from this dataset come with ground-truth poses. Three particularly long and therefore scale drift-prone sequences, namely 00, 02, and 08, were selected for testing, while the remaining eight sequences were used to generate training data.

An individual input to the network is two successive images from a sequence, each 240×120 pixels in size. The dataset contains frames of various sizes, but most are 1241×376 pixels. While simply re-sizing all images to the input size might seem appropriate, it ignores the fact that different sequences have different intrinsic camera calibrations. For this reason, we normalize all images to a single calibration matrix K_{norm} . For each homogeneous pixel coordinate \mathbf{u} in the new normalized image I_{norm} , the corresponding colour in the original image is computed using

$$I_{\text{norm}}(\mathbf{u}) \leftarrow I_{\text{orig}}(K_{\text{orig}} K_{\text{norm}}^{-1} \mathbf{u}), \quad (2)$$

where K_{orig} and I_{orig} are the original calibration matrix and image respectively. Subpixel interpolation is found unnecessary, so we round $(K_{\text{orig}} K_{\text{norm}}^{-1} \mathbf{u})$ to the nearest integer. The target intrinsic calibration matrix K_{norm} has focal

lengths $(f_x, f_y)=(300, 300)$ and principal point $(u_0, v_0) = (120, 60)$.

The sequences used for training contain about 10000 consecutive pairs in total. The following strategies are employed to increase the size of the training set by two orders of magnitude:

- The KITTI dataset supplies stereo images. Images from both cameras are utilized as training input.
- Each training pair of images has their order reversed (as if the camera were moving backwards), while keeping their target value the same. Note that we are predicting speed not velocity, and so a negative speed is not used here.
- Each image from a training pair is used to create a new training pair with the same image twice as input to simulate a stationary camera. The target value for these new pairs is 0.
- Each image in a pair is flipped horizontally, while keeping the pair's target value the same.

These measures increase the size of the dataset to some 1600000 examples. Additionally, to prevent over-fitting during training, dropout is applied to convolutional layers and fully connected layers with a 30% chance.

For the experiments described below, the network was trained using gradient descent, taking some 8 hours to converge using Nvidia Titan X GPU for 1.6 million examples.

3. Using speed in localization and mapping

Using the speed for frame-by-frame visual odometry is straightforward. The essential matrix is computed and decomposed into its five constituent degrees of freedom (*i.e.* three for rotation and two for *direction* of translation) (*c.f.* [15]), and the per-frame magnitude of translation used as the sixth degree of freedom in the Euclidean transformation between camera poses. See [13] for an application.

But using the output in a location and mapping algorithm is not so obvious. In odometry, the scale of the relative translation is estimated in isolation between pairs of frames. In localization and mapping, the scale of the translation is set by the surrounding map. There is no parameter that can simply be rescaled using the speed estimate. Empirically we find the most satisfactory approach is to regularize the usual minimization of reprojection error by imposing constraints between keyframes that involve the speed estimates from the convolutional network.

Our modified adjustment finds the set of landmark positions $\mathbf{X}_{jw} \in \mathcal{X}$ and keyframe poses $\mathbf{T}_i \in \mathcal{T}$ that minimize a regularized cost

$$\{\mathcal{X}, \mathcal{T}\}^* = \arg \min_{\{\mathcal{X}, \mathcal{T}\}} [\mathcal{E}_r(\mathcal{X}, \mathcal{T}) + \lambda \mathcal{E}_c(\mathcal{T})]. \quad (3)$$

The image reprojection cost is the standard expression

$$\mathcal{E}_r = \sum_{\mathbf{X}_{jw} \in \mathcal{X}} \sum_{\mathbf{T}_j \in \mathcal{T}} \mathbf{r}_{ij}^\top \mathbf{W}_{ij}^{-1} \mathbf{r}_{ij}, \quad (4)$$

where the residual between measured and predicted projection of landmark \mathbf{X}_{jw} in keyframe \mathbf{T}_i is

$$\mathbf{r}_{ij} = (\mathbf{x}_{ij} - \hat{\mathbf{x}}(\mathbf{T}_i, \mathbf{X}_{jw})) \quad (5)$$

and \mathbf{W}_{ij}^{-1} is the geometric error covariance.

The constraint cost places pose restrictions between keyframes, much as in pose-graph optimization. That is,

$$\mathcal{E}_c = \sum_{\mathbf{T}_k, \mathbf{T}_l \in \mathcal{T}, k \neq l} \mathbf{R}_{kl}^\top \mathbf{R}_{kl} \quad (6)$$

where \mathbf{R}_{kl} denotes the pose graph-residual [34]

$$\mathbf{R}_{kl} = \log_{SE(3)}(\mathbf{T}_k \mathbf{T}_l^{-1}) \quad (7)$$

between keyframes k and l with poses \mathbf{T}_k and \mathbf{T}_l , where \mathbf{T}_{kl} is the relative pose, and $\log_{SE(3)}(\cdot)$ is the inverse of the exponential map. When the pose constraint \mathbf{T}_{kl} is adhered to then $\mathbf{T}_k \mathbf{T}_l^{-1} = \mathbf{T}_{kl}^{-1}$ and the error is $\mathbf{R}_{kl} = \log_{SE(3)}(\mathbf{I}) = \mathbf{0}$.

More typically, pose-graph constraints are used for closing loops and are deployed either in an optimisation that is distinct from bundle adjustment, or in a joint optimisation that operates on a separate part of the map *e.g.* [33]. In our work the constraints are imposed within the bundle adjustment itself. As we are only concerned with penalizing deviations from the CNN's predicted speed rather than a full rigid-body constraint, the constraints are initialized as

$$\mathbf{T}_{kl} = \mathbf{T}_l^{-1} \mathbf{T}_k. \quad (8)$$

At this point, the constraint error \mathcal{E}_c is zero. To incorporate the raw speed predictions from the network, the speed of the camera s_{kl} is obtained from the keyframes' images I_k and I_l . The translational component \mathbf{t}_{kl} of the constraint \mathbf{T}_{kl} is then re-scaled by the speed estimate of the network

$$\mathbf{t}_{kl} \leftarrow s_{kl} (\mathbf{t}_{kl} / |\mathbf{t}_{kl}|). \quad (9)$$

It is worth emphasizing a difference between our constraints and those used to adjust scale drift at loop closure in [34]. While the latter penalize the difference in the relative scale of the map between keyframes at loop-closure, ours penalize differences in the estimated and measured speed of the camera during an open-loop trajectory.

3.1. Implementation overview

The recovery of speeds and their use in regularizing constraints can be implemented in any keyframe-based monocular localization and mapping system. In this paper we use an in-house method that operates by maintaining a

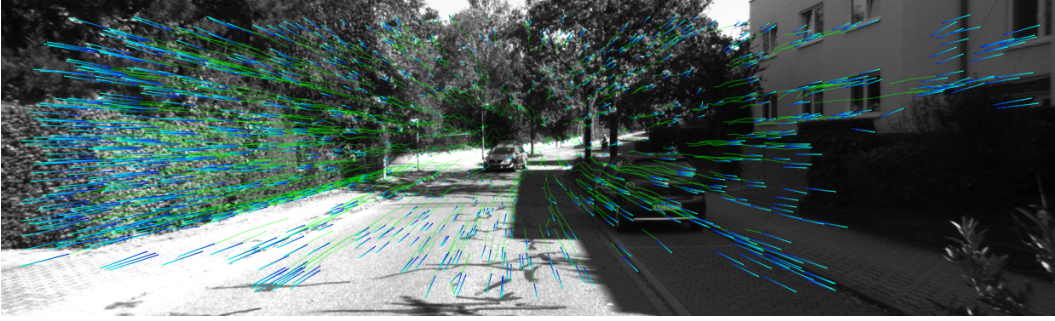


Figure 3. Features and feature tracks.

constant-sized set of feature tracks as its core representation. Fig. 3 shows an example of matches and tracks used in the method.

To maintain feature tracks, our method performs the following steps. In the very first frame captured a set of corners is extracted [29] to initialize the tracks, here limited in number to 1000. For every new frame, tracks are propagated forward from their last seen position using KLT-based feature tracking [3] and validated by reverse matching. If the number of tracks falls below the 1000 threshold new corners are extracted and tracks initiated to deal with the shortfall. Feature tracks are then used for data association throughout processing.

The 3D map is initialized using standard fundamental matrix decomposition using matches between two initial keyframes. Seeking common feature-tracks between any two frames yields matches between them. New landmarks are triangulated from matches, and then associated with tracks.

When estimating the pose of the current camera frame, landmark measurements may be obtained simply by seeking active feature tracks with an associated landmark. The initial pose of the frame is set using a constant velocity motion model, and refined by minimizing reprojection error.

New keyframes are added to the map once the distance between the current camera centre and the closest keyframe exceeds a given threshold. In our case, this is the distance between the first two keyframes of the map. Once a new keyframe is inserted into the map, a speed constraint from the CNN is inserted between the current frame and the closest keyframe in the map. A local bundle adjustment using our proposed speed-constraints is then performed. A window of the 10 nearest keyframes is used.

4. Evaluation

Using the output of a CNN to stabilize scale in a live localization and mapping method imposes two absolute requirements on the network’s output. These are that (i) that the CNN delivers timely results and (ii) the speed is within a consistent factor of the ground truth. There is also

a further desideratum: that the factor is close to unity.

4.1. Timing

The timing requirement is easily met. The latency between the network receiving an image pair and delivering an estimated speed is about 1 ms.

4.2. Consistency

The CNN’s ability to infer speed consistently is tested here using KITTI sequences 00, 02 and 08. We note that while all of these contain loops and therefore offer the possibility of loop-closure, we intentionally do not do so in order to increase the potential for drift. (Before closure, drift scales with distance travelled, but after it scales broadly with Euclidean distance from the origin of coordinates [7].) To measure consistency, we minimize a mean squared deviation from ground truth over all CNN predictions

$$\min_{\alpha} \sum_k (s_k^{\text{cnn}}/\alpha - s_k^{\text{gt}})^2 \quad (10)$$

and then scale all raw speeds by the optimal α .

Fig. 4(a) shows the conditioned values from the CNN compared with ground truth for sequence 00 of the KITTI dataset. The variations in speed are mostly well reproduced. Fig. 4(b) shows the speed recovered after applying the regularized bundle adjustment introduced in Section 3. Note that the output is not a mere smoothing of the raw CNN output, but a speed that is consistent with optimized camera positions in an optimized map. For example, around frame 3200 there is a period when the raw speed is in error, but the bundle adjusted trajectory speed is fully corrected. But note too that unfavourable image conditions can cause both the CNN and bundle adjustment to struggle. For example, after frame 4000 there is a period when the imagery gives poor quality information to both. A pair of images from this period is given as Fig. 8(a).

Fig. 5 plots the *differences* between the recovered and ground truth speeds for KITTI video sequences 00, 02 and 08, each around 450 s duration, and across a number of

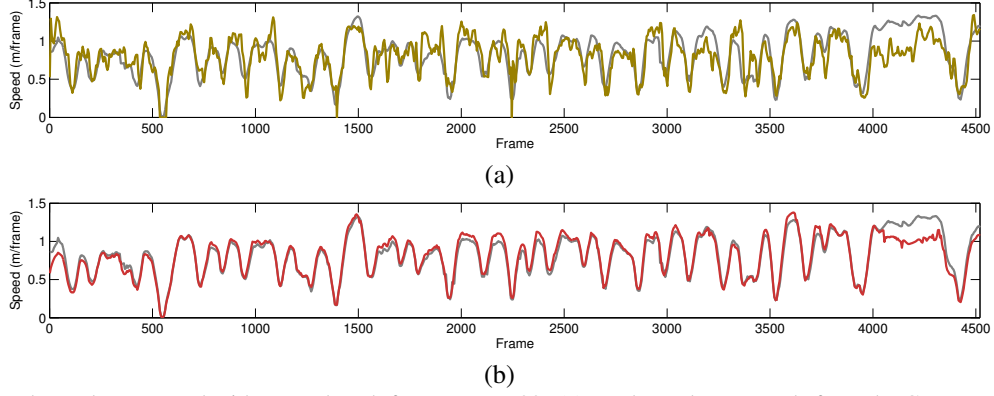


Figure 4. Recovered speeds compared with ground truth for sequence 00. (a) Moderated raw speeds from the CNN, and (b) speeds from the output of regularized bundle adjustment. The ground truth is plotted in grey.

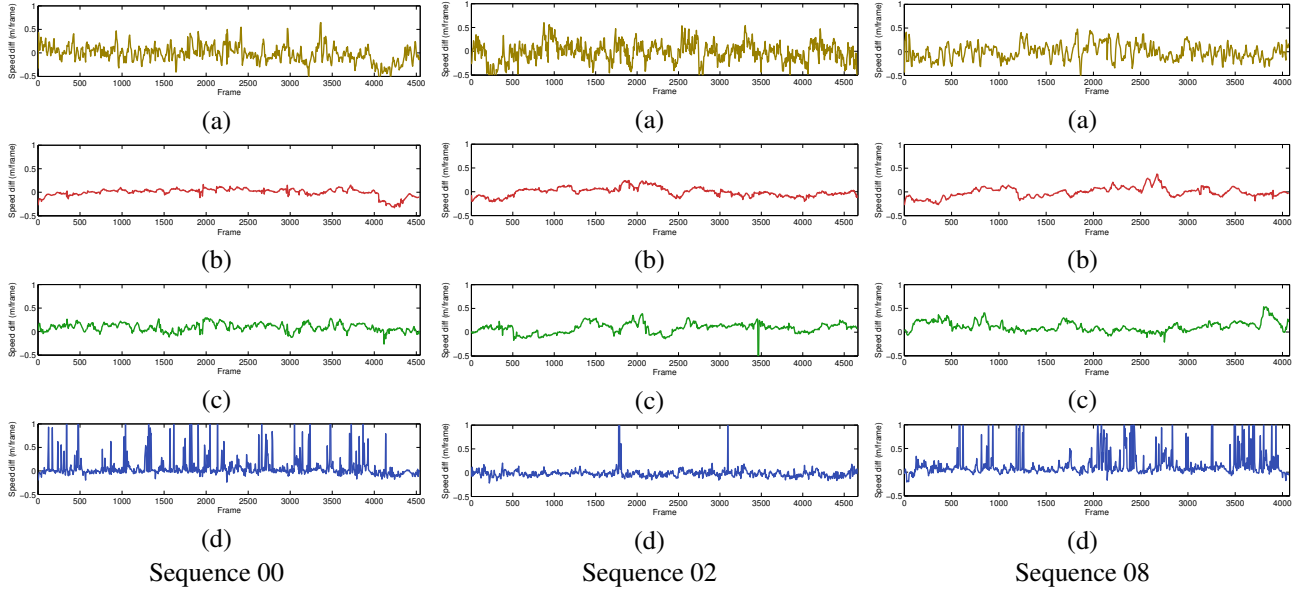


Figure 5. For sequences 00, 02, and 08: the differences between recovered speeds and ground truth speed per frame for speed (a) from the CNN alone; (b) from the CNN and bundle adjustment; (c) from Object supplemented bundle adjustment [11]; (d) from visual odometry using a fixed-height camera assumption [13]. Each plot is from about 450 s of video.

methods. Panel (a) shows the difference for the moderated CNN, and (b) is that for the regularized bundle adjustment. The means and standard deviations of these difference signals are given in Table 1. (For zero mean, the standard deviation in the difference is the rms error between the original signals.) Their values for sequence 00, $(\mu, \sigma) = (-0.014, 0.177)$ and $(\mu, \sigma) = (-0.002, 0.085)$, show a lack of bias in both, and a reduction of an order of magnitude in difference by adjustment. Panel (c) shows the difference for a method which recognizes cars on the roadway and uses the average of their specified lengths to establish scale and hence speed [11]. Using the data in [11] gives $(\mu, \sigma) = (0.092, 0.083)$ for the difference. Last, (d) shows the difference between measured and ground truth speed using an odometry method that uses a fixed camera height as-

sumption [13]. For sequence 00 $(\mu, \sigma) = (0.072, 0.252)$; but the difference is biased and cannot be approximated as normal. Overall, Table 1 indicates that using CNN speeds in a regularized bundle adjustment give the greatest drift stability.

Method	Seq #00		Seq #02		Seq #08	
	μ	σ	μ	σ	μ	σ
(a) CNN alone	-0.014	0.177	-0.018	0.203	-0.004	0.165
(b) CNN & BA	-0.002	0.085	-0.006	0.092	-0.008	0.104
(c) Object BA [11]	0.092	0.083	0.084	0.110	0.119	0.108
(d) Fixed-height [13]	0.072	0.252	-0.012	0.160	0.154	0.349

Table 1. Means and standard deviations of the difference between a recover speed and ground truth, as plotted in Fig. 5.

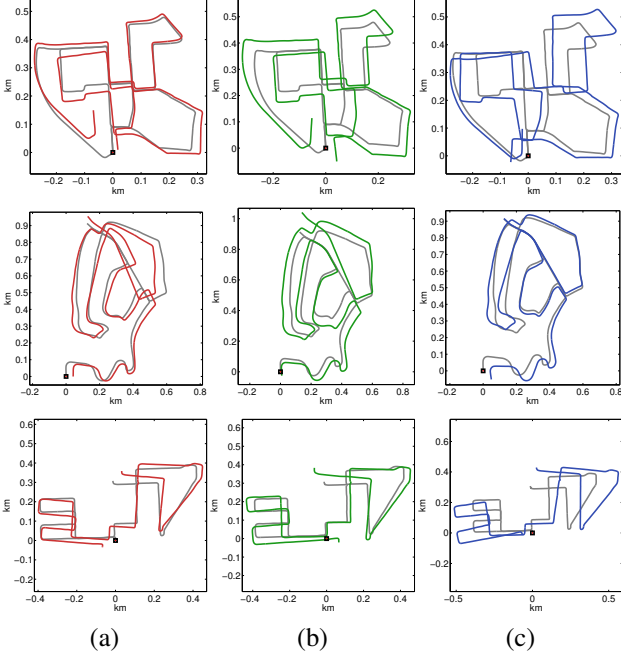


Figure 6. Comparison of estimated trajectories for test sequences (from top to bottom 00, 02 and 08). (a) shows those from our method (b) are from object-supplemented bundle adjustment, and (c) are from odometry using a fixed-camera height constraint. In each, the ground truth is in grey.

Fig. 1 showed a comparison of the estimated camera trajectory of the same localization and mapping method with and without our proposed speed regularization for sequence 00. Without using any speed constraint, some 3000 m of drift accumulates in a trajectory of just under 3000 m in actual length. The speed regularized trajectory drifts by some 100 m. Fig. 6 shows additional estimated trajectories for our proposed method in column (a), but now compared in column (b) with the methods that uses objects to provide scale [11] and in (c) with the fixed height camera constraint [13]. In each case, the CNN regularized bundle adjustment performs best.

4.3. Absolute accuracy of the CNN

The CNN appears highly consistent — it delivers accurate relative speeds which can be mapped onto ground truth with a single scale factor. For absolute speeds, we find it tends to deliver a small over-estimate. Fig. 7 shows a histogram of individual ratios $\alpha_i = s_i^{\text{cnn}} / s_i^{\text{gt}}$ for some 13000 measurements for which we have ground truth. The bins are in $\log_{10}(\alpha)$, so that 0 corresponds to $\alpha=1$ and the under- and over-estimated populations are properly represented. The average log-ratio is found as $\langle \log_{10} \alpha \rangle = 0.0149$, and the corresponding ratio is $\alpha = 1.035$.

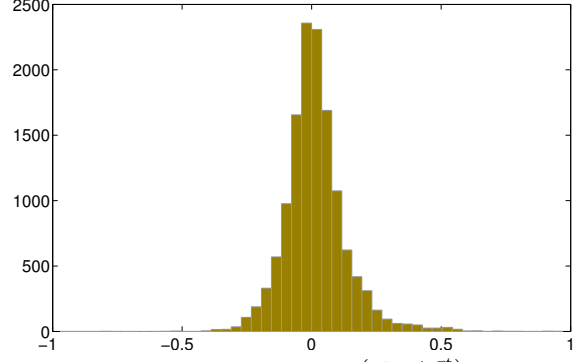


Figure 7. Histogram of individual $\log_{10}(s_i^{\text{cnn}} / s_i^{\text{gt}})$ values, the logs of the speed returned by the CNN to measured ground truth. There is a slight bias towards over-estimation of the speed.

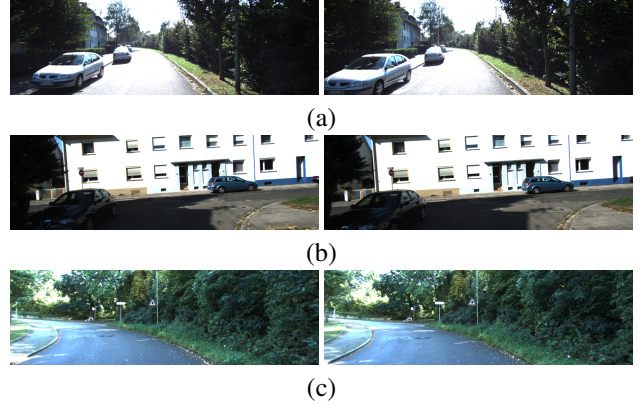


Figure 8. Image pairs that resulted in erroneous speed estimates from the proposed deep network.

4.4. Failure modes

While the network is able to predict camera speed with reasonable accuracy for the majority of video frames, there are some circumstances where this is not the case. Our method is robust against deviations from ground-truth, as its output only serves as one of many constraints in the proposed bundle adjustment in Eq. (3). As a result, the method is able to recover from intermittent errors in speed prediction. This can be seen in the difference in between the raw output and that obtained by our bundle adjustment in Fig. 5.

There are circumstances where the network repeatedly fails to estimate speed correctly. Fig. 8 shows example frames where this has occurred in testing. Inspection of such images shows that repetitive structure, whether from buildings or from foliage, often lead to inaccurate speed estimates.

4.5. Transfer to general handheld motion

To assess the network’s ability to generalise to handheld motion, we utilize a video sequence captured using a handheld phone camera (a Samsung Galaxy S6, pre-calibrated

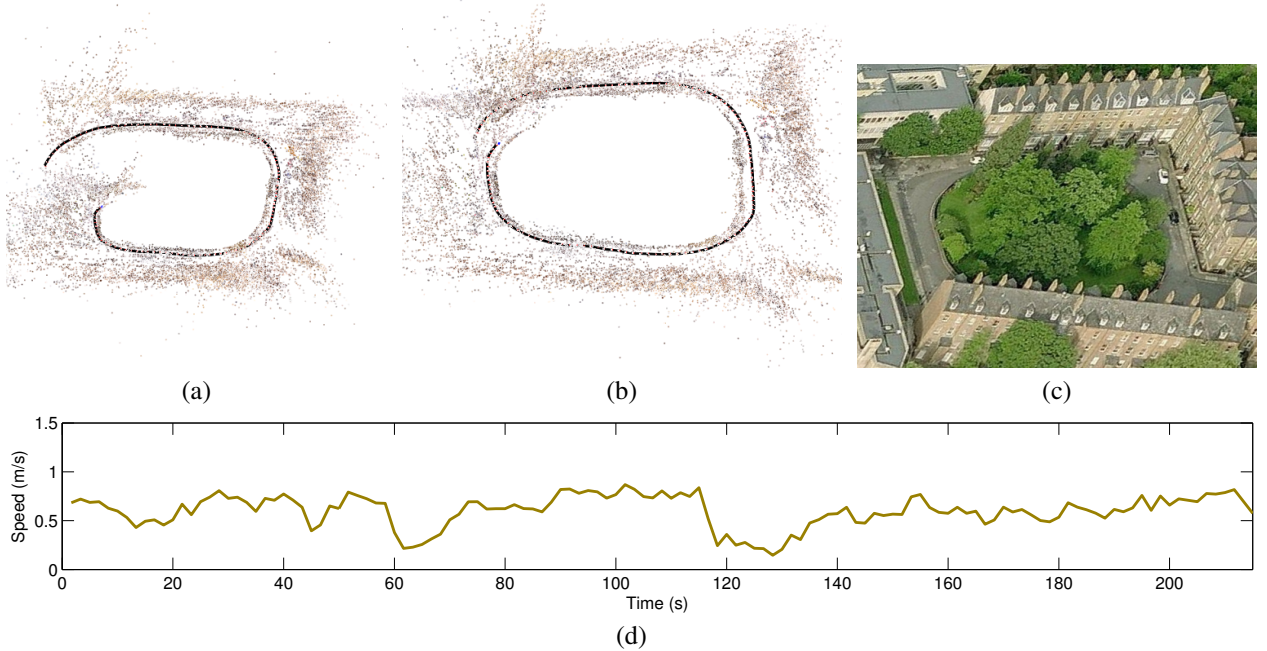


Figure 9. CNN and speed-regularized bundle adjustment applied to imagery from a hand-help camera phone. (a) Without and (b) with constraint. (c) Aerial image. (d) The raw speed in ms^{-1} .

with fixed focal length). The sequence is contained to a small loop and, in the absence of ground truth, performance is evaluated on the qualitative distance between the start and end of the loop which lie at the same position in reality. The sequence was filmed with a frame-rate of 30 fps and keyframes were selected every 50 frames.

Fig. 9(a) shows the estimated camera trajectory and map using our localization and mapping method without additional information from the proposed network, and Fig. 9(b) shows the result regularized by speeds from the network. The beginning and end of the loop are closer in the latter. Fig. 9(d) shows the recovered speed. The average is 0.6 ms^{-1} , whereas the duration and length of the loop indicate that the actual average was close to 0.8 ms^{-1} . We note that very little of the training data was low speed and that the camera was held in the hand and not pointed exclusively in the forward direction.

5. Conclusion

This paper has been concerned with scale-drift stabilization in monocular visual localization and mapping.

We have presented a convolutional neural network for speed estimation from pairs of images, training it from calibrated roadway sequences where the image flow field (were it computed) would be predominantly divergent. In vehicle experiments, the speeds delivered by the network appear accurate covering a range between 0.2 and 1.4 m per frame. We have presented results on the accuracy of the absolute speed returned. There is a small tendency to over-

estimate. Although our principal aim has been to determine relative scale accurately and consistently over time, the absolute scale turns out to be of real value.

We have also described a novel regularizer for a bundle adjustment in a keyframe-based camera localization and landmark mapping algorithm. This uses the speeds from the network to impose constraints within the adjustment similar to those used in pose-graph optimization. We have shown that the stabilized scale has lower overall standard deviation than two competing methods, one which uses object size to stabilize scale and the other which uses a constraint based on fixed camera height.

We have demonstrated the use of CNN and regularized bundle adjustment in the recovery of maps from the KITTI data set and on imagery from a hand-held camera. In the latter, the camera motion was an order of magnitude slower and rather more haphazard than the training exemplars. The absolute speed was under-estimated by the network by some 25%.

The volume of training data used for the network was relatively small, but our results demonstrate that it is commensurate with the network’s use in a quite specific domain, outdoors and moving mainly forward (or back). At present we have no evidence on the adaptability (after appropriate training) of our network architecture to other characteristic camera motions. Given the temporal nature of the data, future research might focus on recurrent networks to take advantage of past images and speed-estimates for current speed estimates.

Acknowledgements

This work is supported by Huawei Technologies Co. Ltd. any partially supported by grant EP/J014990 from the UKs Engineering and Physical Science Research Council.

References

- [1] M. Achtelik, M. Achtelik, S. Weiss, and R. Siegwart. On-board IMU and monocular vision based control for MAVs in unknown in-and outdoor environments. In *Proc 2011 IEEE Int Conf on Robotics and Automation*, pages 3056–3063, 2011. 2
- [2] P. Agrawal, J. Carreira, and J. Malik. Learning to see by moving. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 37–45, 2015. 3
- [3] S. Baker and I. Matthews. Lucas-Kanade 20 years on: A unifying framework. *International Journal of Computer Vision*, 56(3):221–255, 2004. 5
- [4] T. Botterill, S. Mills, and R. D. Green. Correcting scale drift by object recognition in single-camera SLAM. *IEEE Transactions on Cybernetics*, 43(6):1767–1780, 2013. 2
- [5] R. O. Castle, D. J. Gawley, G. Klein, and D. W. Murray. Towards simultaneous recognition, localization and mapping for hand-held and wearable cameras. In *Proc 2007 IEEE Int Conf on Robotics and Automation*, pages 4102–4107, 2007. 2
- [6] M. Cummins and P. Newman. FAB-MAP: Probabilistic localization and mapping in the space of appearance. *International Journal of Robotics Research*, 27(6):647–665, 2008. 2
- [7] A. J. Davison and D. W. Murray. Sequential localisation and map-building using active vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):865–880, July 2002. 2, 5
- [8] D. Eigen, C. Puhrsch, and R. Fergus. Depth map prediction from a single image using a multi-scale deep network. In *Advances in neural information processing systems*, pages 2366–2374, 2014. 2, 3
- [9] J. Engel, J. Stückler, and D. Cremers. Large-scale direct SLAM with stereo cameras. In *Proc IEEE/RSJ Conf on Intelligent Robots and Systems*, pages 1935–1942, 2015. 2
- [10] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. van der Smagt, D. Cremers, and T. Brox. FlowNet: Learning optical flow with convolutional networks. *arXiv preprint arXiv:1504.06852*, 2015. 2
- [11] D. P. Frost, O. Kahler, and D. W. Murray. Object-aware bundle adjustment for correcting monocular scale drift. In *Proc 2016 IEEE Int Conf on Robotics and Automation*, pages 4770–4776, 2016. 1, 2, 6, 7
- [12] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. *International Journal of Robotics Research*, 32(11):1231–1237, 2013. 3
- [13] A. Geiger, J. Ziegler, and C. Stiller. Stereoscan: Dense 3D reconstruction in real-time. In *Proc Intelligent Vehicles Symposium IV*, pages 963–968, 2011. 1, 2, 4, 6, 7
- [14] J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. In *Proc IEEE Int Symp on Computational Intelligence in Robotics and Automation*, pages 318–325, 1999. 2
- [15] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2000. 4
- [16] C. Hide, T. Botterill, and M. Andreotti. Vision-aided IMU for handheld pedestrian navigation. In *Proceedings of the institute of navigation GNSS 2010 Conference, Portland, Oregon*, 2010. 2
- [17] D. Jayaraman and K. Grauman. Learning image representations tied to ego-motion. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1413–1421, 2015. 3
- [18] S.-H. Jung and C. J. Taylor. Camera trajectory estimation using inertial sensor measurements and structure from motion results. In *Proc 15th IEEE Conf on Computer Vision and Pattern Recognition*, pages II:732–737, 2001. 2
- [19] B. Kitt, A. Geiger, and H. Lategahn. Visual odometry based on stereo image sequences with RANSAC-based outlier rejection scheme. In *Proc Intelligent Vehicles Symposium III*, pages 486–492, 2010. 2, 3
- [20] G. Klein and D. W. Murray. Parallel tracking and mapping for small AR workspaces. In *Proc 6th IEEE/ACM Int Symp on Mixed and Augmented Reality*, pages 225–234, 2007. 2
- [21] K. Konda and R. Memisevic. Learning visual odometry with a convolutional network. In *Proc 10th Int Conf on Computer Vision Theory and Applications*, pages I:486–490, 2015. 2
- [22] F. Liu, C. Shen, G. Lin, and I. D. Reid. Learning depth from single monocular images using deep convolutional neural fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(10):2024–2039, 2016. 2
- [23] J. Michels, A. Saxena, and A. Y. Ng. High speed obstacle avoidance using monocular vision and reinforcement learning. In *Proc 22nd ACM Int Conf on Machine Learning*, pages 593–600, 2005. 2
- [24] G. Nützi, S. Weiss, D. Scaramuzza, and R. Siegwart. Fusion of IMU and vision for absolute scale estimation in monocular SLAM. *Journal of Intelligent and Robotic Systems*, 61(1-4):287–299, 2011. 2
- [25] A. Saxena, S. H. Chung, and A. Y. Ng. Learning depth from single monocular images. In *Advances in Neural Information Processing Systems*, pages 1161–1168, 2005. 2
- [26] A. Saxena, M. Sun, and A. Y. Ng. Make3d: Learning 3D scene structure from a single still image. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(5):824–840, 2009. 2
- [27] D. Scaramuzza, F. Fraundorfer, M. Pollefeys, and R. Siegwart. Absolute scale in structure from motion from a single vehicle mounted camera by exploiting nonholonomic constraints. In *Proc 12th IEEE Int Conf on Computer Vision*, pages 1413–1419. IEEE, 2009. 2
- [28] S. Se, D. G. Lowe, and J. J. Little. Local and global localization for mobile robots using visual landmarks. *Proc IEEE/RSJ Conf on Intelligent Robots and Systems, Lausanne, Switzerland, Oct 2-4, 2002*, pages 414–420, 2002. 2

- [29] J. Shi and C. Tomasi. Good features to track. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 593–600. IEEE, 1994. 5
- [30] G. Sibley, C. Mei, I. Reid, and P. Newman. Vast scale outdoor navigation using adaptive relative bundle adjustment. *International Journal of Robotic Research*, 29(8):958–980, 2010. 2
- [31] S. Song and M. Chandraker. Robust scale estimation in real-time monocular SFM for autonomous driving. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1566–1573. IEEE, 2014. 2
- [32] H. Strasdat. *Local accuracy and global consistency for efficient visual SLAM*. PhD thesis, Imperial College, London, 2012. 1
- [33] H. Strasdat, A. J. Davison, J. M. M. Montiel, and K. Konolige. Double window optimisation for constant time visual SLAM. In *Proc 13th IEEE Int Conf on Computer Vision*, pages 2352–2359. IEEE, 2011. 4
- [34] H. Strasdat, J. M. M. Montiel, and A. J. Davison. Scale drift aware large scale monocular SLAM. In *Proc Robotics: Science and Systems Conference*, 2010. 4
- [35] S. Xie, R. Girshick, P. Dollr, Z. Tu, and K. He. Aggregated residual transformations for deep neural networks. *arXiv preprint arXiv:1611.05431*, 2016. 3
- [36] S. Zagoruyko and N. Komodakis. Learning to compare image patches via convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4353–4361, 2015. 3
- [37] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe. Unsupervised learning of depth and ego-motion from video. In *CVPR*, 2017. 3