Dense, Auto-Calibrating Visual Odometry from a Downward-Looking Camera

Jacek Zienkiewicz
j.zienkiewicz12@imperial.ac.uk
Robert Lukierski
r.lukierski12@imperial.ac.uk
Andrew Davison
a.davison@imperial.ac.uk

Department of Computing
Imperial College London
London, UK

Abstract

We present a technique whereby a single camera can be used as a high precision visual odometry sensor in a range of practical settings using simple, computationally efficient techniques. Taking advantage of the local planarity of common floor surfaces, we use real-time dense alignment of a 30Hz video stream as the camera looks down from a fast-moving robot, making use of the whole texture available rather than sparse feature points. Our key novelty, and crucial to the practicality of this approach, is rapid and automatic calibration for 6DoF camera extrinsics relative to the robot frame. Our experiments show robust performance over a range of low-textured real surfaces.

1 Introduction

Computer vision is now widely considered a suitable technology for mobile robot navigation, but few practical mass-market solutions rely solely on cameras for outward-looking sensing. In this paper we focus on Visual Odometry (VO), or incremental camera-based motion estimation. The slip-induced errors of wheel odometry are well known, and are acute in the case of light-weight, low-cost robots moving on unpredictable surfaces such as carpet. VO eliminates these problems, but most VO are heavyweight, complicated systems which reconstruct and track the world in full 3D in the form of a point cloud. Here, we instead take direct advantage of the domain knowledge that wheeled mobile robots move on globally or locally planar surfaces. We point a standard single camera down at the floor, and estimate motion by tracking the natural floor texture using whole image dense alignment. By using every image pixel rather than sparse features to contribute to a global motion estimate, we find that almost every surface has trackable texture when observed from close enough, and we are able to show robust and accurate real-time performance on a range of normal poorly-textured surfaces such as carpet, vinyl and wood.

An important aspect to ensure the practicality of a VO system is calibration; in order to achieve unbiased motion estimation it is crucial that the position of the camera relative to the robot is known accurately. We therefore present an accurate, rapid and fully automatic auto-calibration procedure for the extrinsics of a downward-looking camera relative to the

robot frame. We first auto-calibrate for the out-of-plane camera angles which are uniquely consistent with planar motion. We then show that if a simple external source of relative positioning information is available (such as wheel odometry in good conditions), we can estimate the full metric 6DoF extrinsic camera pose in the robot frame of reference.

Our real-time vision implementation is implemented almost entirely in parallel and currently relies on a laptop PC GPU but looking ahead to the low-cost and low-power parallel processors that will be available in embedded platforms in the near future.

2 Related Work

2.1 Visual odometry

Motion estimation using cameras has a long history, but the term 'visual odometry' was to our knowledge coined by Nistér et al. [15], clarifying the important class of problems where accurate but purely incremental motion estimation can usefully be provided by a camera system. Since that time there have been several notable VO systems for general 6DoF motion; the most successful relying on stereo vision (e.g. the incremental parts of [7, 12]).

VO becomes easier if domain assumptions can be brought strongly into play, and other authors have considered the case of planar robot motion over a ground surface. As in our work, Campbell et al. [2] demonstrated how a single camera can track floor texture, but they used optical flow computation and with a partial forward-looking view to estimate orientation. In their system, which used feature correspondences, Kitt et al. [8] argued convincingly for taking advantage of the prior knowledge that their camera was viewing a planar floor.

However, none of these authors went all the way in making the best possible use of the planar scene assumption. Used properly it allows *all the pixels* in a video sequence to contribute to motion estimation via iterative dense alignment. Lovegrove et al. [10] showed the power of this approach in the specific application of on-road vehicle motion estimation from a rear parking camera by observing the planar road surface. We adopt the core approach of that work, but now bring it to the domain of a small indoor robot which drives and rotates rapidly over a variety of real-world surfaces with different texture characteristics. Importantly, unlike the manual extrinsic calibration used by Lovegrove et al. [10] and the other authors mentioned above we show that a planar VO system can be rapidly auto-calibrated from a short sequence without the need for any special markers or targets. We demonstrate experimentally that unbiased and robust VO is obtained from our system over the full range of dynamics of our experimental robot's motion, and in particular highlight the ability of our VO to measure precise local motion at sub-millimetre resolution.

2.2 Extrinsic Auto-Calibration of Sensors

Our method for auto-calibration of the robot-camera configuration merits comparison with other recent work on auto-calibrating the extrinsics of outward-looking sensors on mobile robots. Most often this is achieved either with reference to a carefully-engineered ground truth positioning system or against a full SLAM system in what is sometimes called SCLAM, Simultaneous Calibration, Localisation and Mapping [3, 9]. Such methods are impressive but heavyweight (with the normal complications associated with SLAM) compared to our method, which has more in common with the recent work by Brookshire and Teller [1] who

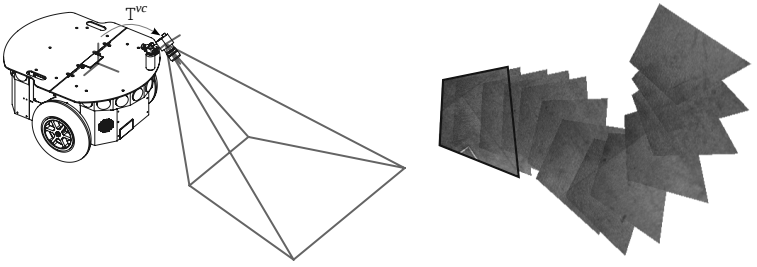

Figure 1: Left: Robot and camera set-up; the camera view frustum intersects the ground plane. Right: Keyframes registered into a local mosaic as part of the auto-calibration process.

demonstrated that external incremental pose measurements are sufficient to recover sensor calibration provided that degenerate trajectories are avoided.

3 Dense VO from a Downward-Looking Camera

Visual odometry in our method uses a whole image alignment approach evolved from the iterative technique introduced by Lucas and Kanade [11] to estimate the incremental planar motion of a robot, which is consistent with the image motion observed by a single camera at the front of the robot. Fig. 1 shows the essential geometry of our method.

3.1 Camera and Robot Geometry

Let $\mathtt{T}^{v_l v_r}$ be the transformation describing the relative motion of the robot between time-steps when its camera captures two images: a live image $\mathcal{I}^l$ and an overlapping earlier reference image $\mathcal{I}^r$. We assume that the camera is bolted to the robot in an arbitrary but fixed location, described by transformation $\mathtt{T}^{vc}$. Since $\mathtt{T}^{vc}$ is constant, the relative motion of the camera between the two images in its own frame of reference, $\mathtt{T}^{lr}$, is related to the motion of the robot $\mathtt{T}^{v_l v_r}$ and can be parameterised by planar motion of the robot $\mathbf{x} \in \mathfrak{se}(2)$ as follows:

$$\mathtt{T}^{lr}(\mathbf{x}) = \mathtt{T}^{cv}\mathtt{T}^{v_l v_r}(\mathbf{x})\mathtt{T}^{vc} \,. \tag{1}$$

In this case, the vector $\mathbf{x}$ is equivalent to the vector $(x, y, \theta)^\top$.

3.2 Homography Based Visual Odometry

Our two images (live $\mathcal{I}^l$ and reference $\mathcal{I}^r$) of the same ground plane are related by a plane-induced homography $\mathtt{H}^{lr}$. This homography transforms pixel coordinates in the reference image into the pixels in the live image and depends on the camera motion $\mathtt{T}^{lr} = (\mathtt{R}^{lr} \mid \mathbf{t}^{lr})$ and the parameters $(\mathbf{n}^\top, d)$ of the plane in the camera r frame of reference [5, p. 327]:

$$\mathtt{H}^{lr} = \mathtt{K}\mathtt{T}^{lr}(I \mid -\mathbf{n}_{d_c})^\top \mathtt{K}^{-1} \,. \tag{2}$$

Here $\mathbf{n}_{d_c} = \frac{\mathbf{n}}{d}$, where $\mathbf{n}$ is the unit vector normal to the plane, and d is the perpendicular distance from the plane to the centre of the camera. $\mathtt{K}$ is the camera intrisic calibration matrix. Using Eq. 1 we can parameterise the homography in terms of the robot's motion $\mathtt{T}^{v_l v_r}$:

$$\mathtt{H}^{lr}(\mathbf{x}) = \mathtt{K}\mathtt{T}^{cv}\mathtt{T}^{v_l v_r}(\mathbf{x})\mathtt{T}^{vc}(I \mid -\mathbf{n}_{d_c})^\top \mathtt{K}^{-1} \,. \tag{3}$$

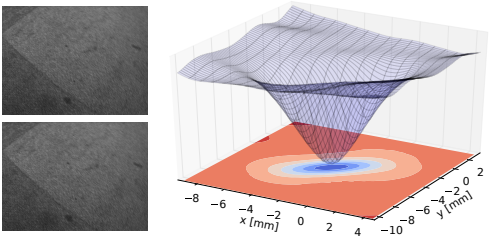

Figure 2: Typical dense tracking cost function. The function depends on the three degrees of freedom of planar robot motion, but only a translational slice is shown here.

The goal of VO is to find a robot motion $\mathbf{x}$ that best registers $\mathcal{I}^l$ and $\mathcal{I}^r$ using homography $\mathtt{H}^{lr}$. Let us first assume that the parameters of the plane $(\mathbf{n}^\top, d)$, camera pose $\mathtt{T}^{vc}$ and camera intrinsics $\mathtt{K}$ are known and only the robot motion needs to be estimated. Determining camera intrinsics is a straightforward, one-off procedure for a certain sensor and is out of the scope of this work. A method for auto-calibrating $(\mathbf{n}^\top, d)$, and $\mathtt{T}^{vc}$ will be presented in Section 4.

3.3 Dense Tracking via Iterative Optimisation

In order to find the motion of the robot $\mathbf{x}$, and thus the homography $\mathtt{H}^{lr}$ we transform all pixels in the live image via the current motion estimate into the reference image, and define an energy function F which measures the discrepancy across the whole image in the form of the sum of squared differences between all pairs of superposed pixels:

$$F(\mathbf{x}) = \frac{1}{2}\sum_{\mathbf{p}} (f_{\mathbf{p}}(\mathbf{x}))^2 = \frac{1}{2}\|\mathcal{I}^l(\pi(\mathtt{H}^{lr}(\mathbf{x})\mathbf{p})) - \mathcal{I}^r(\pi(\mathbf{p}))\|_2^2 \,. \tag{4}$$

Here $\mathbf{p}$ are homogeneous image coordinates and π is a function that performs homogeneous projection, whereas the function $f_{\mathbf{p}}(\mathbf{x})$ measures per pixel difference. An example plot of the cost function with respect to the translational degrees of freedom of the robot can be seen in Fig. 2. In order to minimise $F(\mathbf{x})$ we perform an iterative optimisation based on Efficient Second-order Minimisation (ESM) [12, 13], initialising the value of $\mathbf{x}$ to the value found on the previous VO timestep. This approach requires us to evaluate the partial derivatives of the cost function at $\mathbf{0}$ and at the solution $\mathbf{x} = \mathbf{x}_0$. This can be computed by applying the following update rule:

$$\hat{\mathtt{T}}^{v_l v_r} \leftarrow \hat{\mathtt{T}}^{v_l v_r}\mathtt{T}(\hat{\mathbf{x}}) \,, \tag{5}$$

where $\hat{\mathtt{T}}^{v_l v_r}$ is the current estimate of the solution, and $\mathtt{T}(\hat{\mathbf{x}})$ is a small update to the estimate parameterised by $\hat{\mathbf{x}} \in \mathfrak{se}(2)$, and using the gradients of the live and reference images to calculate per pixel partial derivatives, as shown in [10]:

$$\frac{\partial f_{\mathbf{p}}(\mathbf{x})}{\partial x_i} = \frac{\partial \mathcal{I}^l(\mathbf{a})}{\partial \mathbf{a}}\bigg|_{\mathbf{a}=\pi(\mathtt{H}^{lr}(\mathbf{x})\mathbf{p})} \frac{\partial \pi(\mathbf{b})}{\partial \mathbf{b}}\bigg|_{\mathbf{b}=\mathtt{H}^{lr}(\mathbf{x})\mathbf{p}} \frac{\partial \mathtt{H}^{lr}(\mathbf{x})}{\partial x_i}\mathbf{p} \tag{6}$$

$$\frac{\partial \mathtt{H}^{lr}(\mathbf{x})}{\partial x_i} = \mathtt{K}\mathtt{T}^{cv}\hat{\mathtt{T}}^{v_l v_r}\frac{\partial \mathtt{T}(\mathbf{x})}{\partial x_i}\mathtt{T}^{vc}(\boldsymbol{I}|-\mathbf{n}_{d_c})^\top \mathtt{K}^{-1} \,, \tag{7}$$

where $\frac{\partial \mathtt{T}(\mathbf{x})}{\partial x_i}\Big|_{\mathbf{x}=\mathbf{0}} = \mathtt{G}_i^{\mathbf{SE}(2)}$ is the ith generator of the $\mathbf{SE}(2)$ group.

4 Auto-Calibration

In Section 3 we assumed that the pose of the camera with respect to the robot frame of reference and the normal of the plane were known. We will show now how these parameters can be automatically calibrated using a short image sequence from the moving robot without the need for targets or markers. This procedure has two main steps. First, based purely on the image sequence and the knowledge that the sequence is obtained from planar motion, we can determine the normal of the ground plane relative to the camera, which is equivalent to recovering its elevation and roll angles. Second, we use relative motion estimates from an external source to calibrate for the remaining parameters, which describe the position of the camera relative to the robot: its yaw angle and metric 3DoF translational position.

4.1 Plane normal

When tracking the relative (up-to-scale) incremental ego-motion of the camera, the position of the camera on the robot is irrelevant, and only the orientation of the camera with respect to the ground and the normal of the plane is required. In order to be able to use the ESM described in Section 3.3, we need to find a parametrisation that allows us to calculate per pixel partial derivatives with respect to the robot motion (Eq. 7) and the parameters of the plane. Unlike Silveira et al. [16] we do not use inverse depth to estimate the normal of the plane, but instead exploit the fact that in our set-up the plane unit normal vector $\mathbf{n}$ is defined by the camera orientation $\mathtt{R}^{cv}$,

$$\mathbf{n} = \mathtt{R}^{cv}\,(0,0,1)^\top \ . \tag{8}$$

and as the camera moves in the plane its elevation and roll angles do not change.

Since in Eq. 1, $\mathtt{T}^{cv} = (\mathtt{R}^{cv} \mid \mathbf{t}^{cv})$, where $\mathtt{R}^{cv} \in \mathbf{SO}(3)$, $\mathbf{t}^{cv} = (0,0,1)^\top$, the homography (Eq. 3) will depend on two sets of parameters: the planar motion from frame to frame $\mathbf{x}$, and the orientation of the camera with respect to the plane $\mathbf{y} = (y_1, y_2, y_3) \in \mathfrak{so}(3)$ (note that there is still an ambiguity in the yaw angle, therefore we constrain $y_3 = 0$):

$$\mathtt{H}^{lr}(\mathbf{x},\mathbf{y}) = \mathtt{K}\mathtt{T}^{cv}(\mathbf{y})\mathtt{T}^{v_l v_r}(\mathbf{x})\mathtt{T}^{vc}(\mathbf{y})(I|-\mathbf{n}(\mathbf{y}))^\top \mathtt{K}^{-1} \ . \tag{9}$$

By rewriting parts of Eq. 9 as follows:

$$\mathtt{T}^{cv}(\mathbf{y}) = \mathtt{R}^{cv}(\mathbf{y})\mathtt{M} \qquad \text{and} \qquad \mathtt{T}^{vc}(\mathbf{y})(I|-\mathbf{n}(\mathbf{y}))^\top = \mathtt{N}\mathtt{R}^{vc}(\mathbf{y}) \ , \tag{10}$$

where

$$\mathtt{M} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & -1 \end{bmatrix} \qquad \text{and} \qquad \mathtt{N} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & -1 \end{bmatrix} \ , \tag{11}$$

we arrive at the expression for the homography

$$\mathtt{H}^{lr}(\mathbf{x},\mathbf{y}) = \mathtt{K}\mathtt{R}^{cv}(\mathbf{y})\mathtt{M}\mathtt{T}^{v_l v_r}(\mathbf{x})\mathtt{N}\,\mathtt{R}^{vc}(\mathbf{y})\mathtt{K}^{-1} \ , \tag{12}$$

which makes it possible to use the following update rule for $\hat{\mathbf{y}} \in \mathfrak{so}(3)$

$$\hat{\mathtt{R}}^{cv} \leftarrow \hat{\mathtt{R}}^{cv}\mathtt{R}(\hat{\mathbf{y}}) \ . \tag{13}$$

Substituting 13 into 12, and using the fact that $\mathtt{R}^{vc} = (\mathtt{R}^{cv})^{-1}$ eventually leads to

$$\mathtt{H}^{lr}(\mathbf{x},\mathbf{y}) = \mathtt{K}\hat{\mathtt{R}}^{cv}\mathtt{R}(\mathbf{y})\mathtt{M}\hat{\mathtt{T}}^{v_l v_r}\mathtt{T}(\mathbf{x})\mathtt{N}(\mathtt{R}(\mathbf{y}))^{-1}(\hat{\mathtt{R}}^{cv})^{-1}\mathtt{K}^{-1} \ . \tag{14}$$

Since $(\mathtt{R}(\mathbf{y}))^{-1} = \mathtt{R}(-\mathbf{y})$, we arrive at the final expression for the homography:

$$\mathtt{H}^{lr}(\mathbf{x},\mathbf{y}) = \mathtt{K}\hat{\mathtt{R}}^{cv}\mathtt{R}(\mathbf{y})\mathtt{M}\hat{\mathtt{T}}^{v_l v_r}\mathtt{T}(\mathbf{x})\mathtt{N}\mathtt{R}(-\mathbf{y})\hat{\mathtt{R}}^{vc}\mathtt{K}^{-1} \,. \tag{15}$$

In order to calculate the per pixel partial difference in a similar fashion to Eq. 6 we now only need to derive the expressions for the partial derivatives of $\mathtt{H}^{lr}$. With respect to $\mathbf{x}$ we have:

$$\left.\frac{\partial \mathtt{H}^{lr}(\mathbf{x},\mathbf{y})}{\partial x_i}\right|_{\mathbf{y}=\mathbf{0}} = \mathtt{K}\hat{\mathtt{R}}^{cv}\mathtt{M}\hat{\mathtt{T}}^{v_l v_r}\frac{\partial \mathtt{T}(\mathbf{x})}{\partial x_i}\mathtt{N}\hat{\mathtt{R}}^{vc}\mathtt{K}^{-1} \,, \tag{16}$$

and for $\mathbf{y}$:

$$\left.\frac{\partial \mathtt{H}^{lr}(\mathbf{x},\mathbf{y})}{\partial y_i}\right|_{\mathbf{x}=\mathbf{0}} = \mathtt{K}\hat{\mathtt{R}}^{cv}\frac{\partial \mathtt{R}(\mathbf{y})\mathtt{M}\hat{\mathtt{T}}^{v_l v_r}\mathtt{N}\mathtt{R}(-\mathbf{y})}{\partial y_i}\hat{\mathtt{R}}^{vc}\mathtt{K}^{-1} \,. \tag{17}$$

Using the fact that $\left.\frac{\partial \mathtt{R}(\mathbf{y})}{\partial y_i}\right|_{\mathbf{y}=\mathbf{0}} = \mathtt{G}_i^{\mathbf{SO}(3)}$ and $\left.\frac{\partial \mathtt{R}(-\mathbf{y})}{\partial y_i}\right|_{\mathbf{y}=\mathbf{0}} = -\mathtt{G}_i^{\mathbf{SO}(3)}$, where $\mathtt{G}_i^{\mathbf{SO}(3)}$ is the ith group generator for $\mathbf{SO}(3)$, and applying product rule, we finally obtain:

$$\left.\frac{\partial \mathtt{R}(\mathbf{y})\mathtt{M}\hat{\mathtt{T}}^{v_l v_r}\mathtt{N}\mathtt{R}(-\mathbf{y})}{\partial y_i}\right|_{\mathbf{y}=\mathbf{0}} = \mathtt{G}_i^{\mathbf{SO}(3)}\mathtt{M}\hat{\mathtt{T}}^{v_l v_r}\mathtt{N} - \mathtt{M}\hat{\mathtt{T}}^{v_l v_r}\mathtt{N}\mathtt{G}_i^{\mathbf{SO}(3)} \,. \tag{18}$$

Having arrived at this expression we can now efficiently use the ESM machinery introduced in Section 3.3 to jointly find the frame to frame motion and the unit normal vector.

4.2 Keyframes

Although it is possible to estimate the plane normal from just two images, to better constrain the estimation process we combine multiple frames into a local dense map and jointly estimate the motion between consecutive keyframes and the parameters of the plane normal. An example of a dense local map is shown in Fig. 1. When the overlap between the last keyframe and the current frame falls below a defined threshold we add this frame to the map. We usually maintain maps of between 10 and 50 frames.

4.3 Graph Optimisation against External Reference

The planar motion of the robot is related to the motion of the camera via the rigid body transformation (x_c, y_c, γ), and when the motion of the camera is calculated only up to a scale then also by scale factor d, which is inversely proportional to camera height $z_c = \frac{1}{d}$. In the final step of auto-calibration the goal is to find a set of parameters $\mathbf{z} = (x_c, y_c, d, \gamma)$ that best explains the incremental measurements from VO relative to the corresponding synchronised incremental reference measurements of robot motion; in our case wheel odometry in good conditions. Our formulation is similar to that of [1] and is based only on incremental measurements, but we model the problem as a hyper-graph (as proposed in [2]).

In the hyper-graph the nodes $\mathbf{p}_i = (x_i, y_i, \theta_i)$ represent robot poses along the trajectory, and one additional node $\mathbf{z} = (x_c, y_c, d, \gamma)$ represents the camera configuration. Every time an odometry measurement is available we add a new node to the graph. We assume that camera and wheel odometry are well synchronised, and the jitter and synchronisation errors are negligible. Two consecutive nodes $\mathbf{p}_i$ and $\mathbf{p}_{i+1}$ are connected by two edges, one representing the measurement obtained by wheel odometry $\mathbf{w}_{i,i+1} = (\Delta x_{w_{i,i+1}}, \Delta y_{w_{i,i+1}}, \Delta\theta_{w_{i,i+1}})$ and the second the visual odometry measurement $\mathbf{v}_{i,i+1} = (\Delta x_{v_{i,i+1}}, \Delta y_{v_{i,i+1}}, \Delta\theta_{v_{i,i+1}})$. Visual odometry

edges $\mathbf{v}_{i,i+1}$ are in fact hyper-edges as they are also connected to the camera configuration $\mathbf{z}$. There are no loops in this graph and the graph is fixed by the first node. Variables in the graph are the poses of the robot $\mathbf{p}_i$ and the camera calibration $\mathbf{z}$.

The error function $e_i^{\mathbf{w}}(\mathbf{p}_i, \mathbf{p}_{i+1}, \mathbf{w}_{i,i+1})$ measures how well the parameters $\mathbf{p}_i, \mathbf{p}_{i+1}$ satisfy the constraint arising from the wheel odometry $\mathbf{w}_{i,i+1}$:

$$e_i^{\mathbf{w}}(\mathbf{p}_i, \mathbf{p}_{i+1}, \mathbf{w}_{i,i+1}) = (\mathbf{p}_{i+1} \ominus \mathbf{p}_i) \ominus \mathbf{w}_{i,i+1} \ , \tag{19}$$

where $\ominus$ is the inverse of the usual motion composition operator $\oplus$. Another error function $e_i^{\mathbf{v}}(\mathbf{p}_i, \mathbf{p}_{i+1}, \mathbf{z}, \mathbf{v}_{i,i+1})$ measures how well the parameters $\mathbf{p}_i$, $\mathbf{p}_{i+1}$ and $\mathbf{z}$ satisfy the constraints arising from the visual odometry measurement $\mathbf{v}_{i,i+1}$:

$$e_i^{\mathbf{v}}(\mathbf{p}_i, \mathbf{p}_{i+1}, \mathbf{z}, \mathbf{v}_{i,i+1}) = (\mathbf{p}_{i+1} \ominus \mathbf{p}_i) \oplus f(\mathbf{z}, \mathbf{v}_{i,i+1}) \ , \tag{20}$$

where f is the function that transforms the motion from the camera frame of reference to the robot frame of reference using (x_c, y_c, γ) and the scale d. When the scale is equal to 1 this is equivalent to $f(\mathbf{z}, \mathbf{v}_{i,i+1}) = \ominus\mathbf{z} \oplus \mathbf{v}_{i,i+1} \oplus \mathbf{z}$.

The goal of the graph optimisation is to find a set of parameters $(\mathbf{p}^\star, \mathbf{z}^\star)$ that maximise the likelihood of the observed data. It is equivalent to minimising the function:

$$MLE(\mathbf{p}, \mathbf{z}) = \sum_i {e_i^{\mathbf{v}}}^\top \hat{\Omega}_i^{\mathbf{v}} e_i^{\mathbf{v}} + \sum_i {e_i^{\mathbf{w}}}^\top \Omega_i^{\mathbf{w}} e_i^{\mathbf{w}} \ , \tag{21}$$

where $\Omega_i^{\mathbf{w}}$ is the information matrix of the measurement $\mathbf{w}_{i-1,i}$ and $\hat{\Omega}_i^{\mathbf{v}}$ is the projection of the information matrix $\Omega_i^{\mathbf{v}}$ using the current estimate of $\mathbf{z}$. The least squares estimation on this hyper graph is performed using g^2o [8].

5 Evaluation

A visual odometry system measures incremental motion, and will therefore lead to drifting estimates of absolute motion, with absolute pose error increasing with distance. An appropriate measure of performance is therefore a statistical evaluation of the accuracy of incremental motion estimation. We have used a Point Grey Flea2 Camera capturing greyscale images at 640×480 resolution and 30Hz frame-rate with automatic camera settings for shutter time and gain. Our robot platform is a Pioneer 3 DX with an adjustable rigid camera mount, carrying an NVidia GPU-equipped laptop for real-time vision processing. The mean frame-rate of full resolution processing is about 270–300 FPS on a desktop GTX480 GPU and about 60–65 FPS on a mid-range laptop GT650M. In both cases the main CPU is virtually idle.

5.1 Justification for Evaluation Against Wheel Odometry

Our Pioneer 3 DX robot has apparently very accurate wheel odometry on surfaces with good grip such as short carpet, but to verify this and enable its use as the reference for our VO evaluation, we compared wheel odometry with ground truth from an external overhead calibrated camera system observing a target marker on top of the robot. We see in Fig. 3 that the Pioneer's wheel odometry has no observable bias; and indeed that the robot has much better local motion estimation accuracy. We conclude that on good surfaces, it is appropriate to compare VO against this high quality wheel odometry. This of course does not invalidate our general argument regarding the advantages of VO in practical problems with more basic platforms and low grip. Note that in all evaluations we use different data from that used for extrinsics auto-calibration.

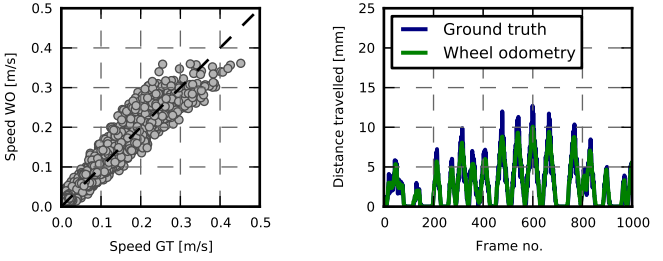

Figure 3: Comparison of ground truth (GT) and the wheel odometry (WO) for incremental motion estimation over 1000 frames of varied motion.

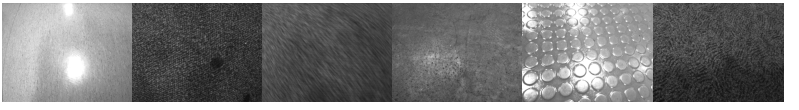

Figure 4: Different surfaces used for evaluation.

5.2 VO Evaluation on Different Surfaces and Conditions

We experimentally evaluated our VO approach against accurate odometry by making extended runs of the robot with different camera heights (70–230mm) on a variety of different surfaces. In each case the camera extrinsics were auto-calibrated using the method described in Section 4, and the robot was driven using a joystick over a trajectory that featured a wide range of motions and turns up to the maximum dynamics of the robot (peak linear velocity 0.8ms^{-1}; peak angular velocity 1.5rads^{-1}). Fig. 4 shows snapshots of the different surface types we tested. Particularly interesting are the shiny, specular surfaces. The non-planar motion of the robot on the surface with raised circular studs resulted in a more noisy motion estimation. The second floor is smooth but also shiny, with strong specularities. Despite this, with no modification, our algorithm gave good results in most cases.

Fig. 5 summarises the results in the form of a histogram. The tight packing of the error around 0 indicates the unbiased estimation of robot motion and a good performance of the algorithm most of the time. The same data, but for only the 130mm camera setting, is presented in another form using a scatter plot, in which each point represents the estimated velocity, both linear and angular, vs. wheel odometry for one frame of the sequence. In the velocity range up to around 0.6ms^{-1} we observe robust performance, but beyond this limit we see some tendency to underestimate the velocity resulting mostly from the reduced inter-frame overlap and motion blur. Usually a failure only happens though when these effects are combined with disadvantageous other issues such as shadows cast by the robot.

5.3 High Precision Incremental Motion

A final experiment investigated and highlighted the capability of downward-looking VO to recover extremely high precision local motion. We mounted the camera at around 4cm from the ground, and drove the robot straight forward at its lowest smooth velocity setting of 2cms^{-1}. The camera observed and tracked a wooden board floor surface on top of which was placed a ruler with clear millimetre markings. By inspecting the image sequence obtained (Fig. 6) we were able confidently to confirm the ground truth motion per frame (at 30Hz) of

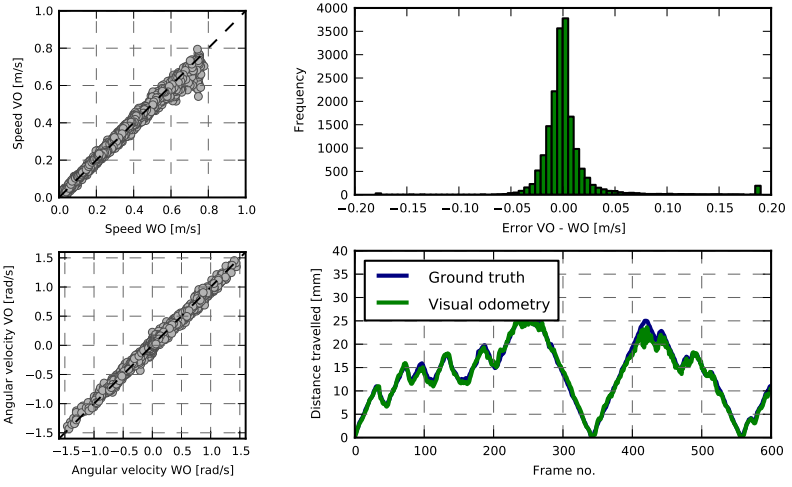

Figure 5: Left: Comparison of the linear speed and angular velocity measured by VO and WO. Upper right: Histogram of the error measured between VO and WO. Lower Right: An example of the estimated per-frame robot linear velocity in units of mm per frame.

around 0.7mm per frame, corresponding to image motion of around 8 pixels per frame.

The results of planar VO applied to this sequence against idealised ground truth can be seen in Fig. 6. We see a remarkable precision in incremental motion estimation, where the per-frame motion is estimated with a standard deviation consistently within 0.1mm. Note that the oscillation observed is due to real vibration of our robot when it first starts moving, and can be seen to damp when a constant velocity is reached.

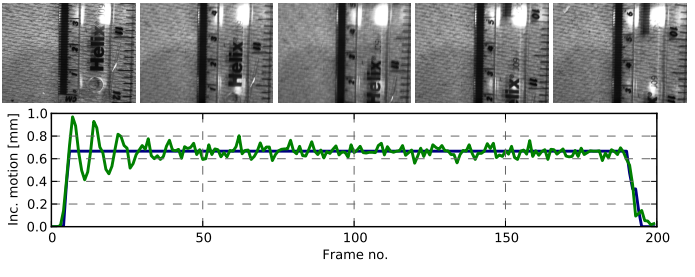

Figure 6: Experiment to investigate ultra-high precision local motion estimation. Top: Images from 10 frame intervals. Bottom: VO against idealised ground truth.

6 Conclusion

We have demonstrated that dense alignment offers a robust and accurate solution for visual odometry which is highly practical thanks to rapid auto-calibration and efficient parallel implementation. Dense alignment methods are highly scalable, and it will be interesting to see the trade-offs in VO when camera frame-rate and resolution are varied [4]. We see

this work as a first step in a dense, auto-calibrated and parallelisable vision approach to robot SLAM and navigation using ad-hoc, low-cost cameras. In near future work we plan to harness the accurate motion estimates from our VO approach to enable the segmentation and reconstruction of the non-planar regions observed by the camera.

Acknowledgments

We are grateful to Dyson Ltd. for funding this research and collaboration on techniques and ideas. We also thank our colleagues at Imperial College London for daily discussions and assistance, in particular Ankur Handa.

References

[1] J. Brookshire and S. Teller. Automatic Calibration of Multiple Coplanar Sensors. In *Robotics: Science and Systems (RSS)*, 2011.

[2] J. Campbell, R. Sukthankar, I. Nourbakhsh, and A. Pahwa. A Robust Visual Odometry and Precipice Detection System Using Consumergrade Monocular Vision. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2005.

[3] A. Censi, L. Marchionni, and G. Oriolo. Simultaneous Maximum-Likelihood Calibration of Odometry and Sensor Parameters. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2008.

[4] A. Handa, R. A. Newcombe, A. Angeli, and A. J. Davison. Real-Time Camera Tracking: When is High Frame-Rate Best? In *Proceedings of the European Conference on Computer Vision (ECCV)*, 2012.

[5] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, second edition, 2004.

[6] B. Kitt, J. Rehder, A. Chambers, M. Schonbein, H. Lategahn, and S. Singh. Monocular Visual Odometry using a Planar Road Model to Solve Scale Ambiguity. In *Proceedings of the European Conference on Mobile Robotics (ECMR)*, 2011.

[7] K. Konolige, M. Agrawal, and J. Solà. Large Scale Visual Odometry for Rough Terrain. In *Proceedings of the International Symposium on Robotics Research (ISRR)*, 2007.

[8] R. Kümmerle, G. Grisetti, H. Strasdat, K. Konolige, and W. Burgard. g^2o: A General Framework for Graph Optimization. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2011.

[9] R. Kümmerle, G. Grisetti, and W.Burgard. Simultaneous Calibration, Localization, and Mapping. In *Proceedings of the IEEE/RSJ Conference on Intelligent Robots and Systems (IROS)*, 2011.

[10] S. J. Lovegrove, A. J. Davison, and J. Ibanez-Guzmán. Accurate Visual Odometry from a Rear Parking Camera. In *Proceedings of the IEEE Intelligent Vehicles Symposium (IV)*, 2011.

[11] B. D. Lucas and T. Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, 1981.

[12] E. Malis. Improving vision-based control using efficient second-order minimization techniques. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*, 2004.

[13] C. Mei, S. Benhimane, E. Malis, and P. Rives. Efficient Homography-Based Tracking and 3-D Reconstruction for Single-Viewpoint Sensors. *IEEE Transactions on Robotics (T-RO)*, 24(6):1352–1364, 2008.

[14] C. Mei, G. Sibley, M. Cummins, P. Newman, and I. Reid. A Constant Time Efficient Stereo SLAM System. In *Proceedings of the British Machine Vision Conference (BMVC)*, 2009.

[15] D. Nistér, O. Naroditsky, and J. Bergen. Visual Odometry. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2004.

[16] G. Silveira, E. Malis, and P. Rives. An Efficient Direct Approach to Visual SLAM. *IEEE Transactions on Robotics (T-RO)*, 24(5):969–979, 2008.