

# The Effect of Different Parameterisations in Incremental Structure from Motion

Lukáš Polok, Vincent Lui, Viorela Ila, Tom Drummond, Robert Mahony  
Brno University of Technology, Monash University, Australian National University  
ipolok@fit.vutbr.cz, {vincent.lui,tom.drummond}@monash.edu, {viorela.ila,robert.mahony}@anu.edu.au

## Abstract

Accurate online estimation of the structure of the environment together with the pose of the robot is an important component to enable autonomous robotic applications. This paper analyses the different parameterisations used in structure from motion (SFM) problem in the context of accuracy and efficiency of the online solutions. Three point parameterisations are compared: Euclidean, inverse depth and inverse distance. At the same time two representations, global and local point coordinates are tested. Different metrics are used to compare the results, camera localisation errors, reprojection errors, execution time as well as a complete analysis on how different parameterisations affect the convergence, system's condition number and the incremental solving are provided. The paper shows that, with the correct parameterisation, efficient globally consistent SFM is possible, which under the assumption of small, bounded number of correspondences performs in constant time in open loop.

## 1 Introduction

The three dimensional representation of the environment is an important component to enable autonomous robotic applications. In order to perform tasks, the robot needs to know where it is and how the environment looks like. For that, it continuously builds and updates its position and the environment representation. Simultaneous localisation and mapping (SLAM) and structure from motion (SFM) are the techniques used for this purpose, both sharing similar mathematical formulations. The only difference appears in the type of measurements which are integrated in the estimation process, SFM being restricted to image based sensing. This paper focuses on the SFM problems, but the presented methods and results directly apply to any type of SLAM problems which integrate image sensing of the environment.

Several methods for processing and tracking every pixel in the image in order to obtain a dense reconstruction of the environment exist, but they apply only to small scale applications [Newcombe and Davison, 2010]. In application such as surveillance, monitoring or any other tasks where the robot operates in large scale environment, sparse reconstruction is the method of choice. The sparse reconstruction is always a good starting point to recover the dense structure in a more efficient way [Pollefeys *et al.*, 2001]. The SFM is used to estimate the camera poses and a sparse number of feature points in the environment. The online reconstruction pipeline starts by extracting salient features and matching them over two or more images. This will produce an initialisation of the camera pose and 3D points. The initialisation is followed by tracking the points and updating their estimation together with the new camera poses in an SFM framework. The SFM minimises the reprojection errors of all the points visible in all the cameras. Assuming Gaussian noise, this problem became a nonlinear least squares NLS, where the variables are the 3D points and the camera poses.

This paper analyses the incremental sparse SFM accuracy and efficiency, while several different design options are being considered. The aim is to find the most adequate parameterisation for incremental processing. A good parameterisation should provide an accurate solution, yet be efficient. At the same time we analyse the system variable affected by the updates under different point parameterisations. This has a great importance in applying incremental optimisation strategies as described in [Polok *et al.*, 2013], and which are proven to be very efficient.

## 2 Existing SFM Approaches

The SFM approaches in the literature can be roughly categorized by the parameterisation they are using. The following chapters introduce work done in two basic categories of parameterization, global and local.

## 2.1 Global Parameterisations

A global parameterisation is the popular choice among the early implementations of SFM or visual SLAM systems. This includes Davison *et al.*'s Extended Kalman Filter (EKF) approach along with Eade and Drummond's [Eade and Drummond, 2006] particle-filter based system. Due to the computational complexity of updating the filter, both systems could only handle a small number of landmarks.

Approaching structure from motion and visual SLAM as a bundle adjustment (BA) allows for more accurate yet efficient solution. The problem is formulated as maximum likelihood estimation over all the points and the camera poses given a set of measurements. Those problems are, in general, denser than the SLAM problem, one point being seen by many cameras, therefore specific techniques need to be applied in order to obtain online results. Separating the structure from the camera poses and solving first the reduced camera poses system and then the structure is the most common approach [Agarwal *et al.*, 2010]. Recently, some techniques consider excluding the structure from the optimisation process and use only the multi view constraints that the structure adds to the problems to find optimal position of the cameras [Indelman and Dellaert, 2015]. These techniques are suitable for robot localisation using cameras, but will not provide a map for of the environment for performing path planning or other tasks.

An improvement in capability of handling the number of landmarks is shown in Klein and Murray's Parallel Tracking and Mapping (PTAM) system [Klein and Murray, 2007]. This is made possible by separating tracking the points and the structure estimation into two asynchronous threads, where tracking is serviced at frame rate whereas structure estimation does not have to run at frame-rate. However, some applications require the map to be available every step, so that the robot can perform its tasks.

## 2.2 Relative Parameterisations

In a relative parameterisation, camera poses are only defined relative to one another. Using a graph structure, the cameras are represented as vertices and an edge between two vertices represents a relative camera pose. In this representation, landmark estimates have to be recovered using graph traversal techniques such as breadth first search or shortest path computation using Dijkstra's algorithm. Some examples of SLAM systems using relative representations are in [Guivant and Nebot, 2001; Bosse *et al.*, 2003; Eade and Drummond, 2008; Konolige and Agrawal, 2008].

Mei *et al.*'s RSLAM [Mei *et al.*, 2010] exploits the relative parameterisation by proposing a relative bundle adjustment technique where both, the landmarks and

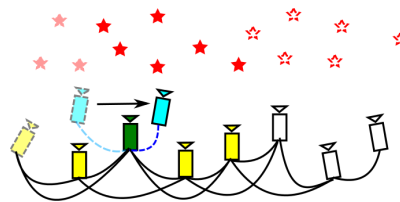


Figure 1: Selection of reference views for visual tracking. The local neighbourhood changes as the current camera's position changes (from dashed cyan to solid cyan). The master view is shown in green and the current neighbour views are shown in solid yellow. The previous neighbour view is shown in translucent yellow.

the camera poses are represented in relative coordinates. This reduces the cost of optimising the map as it allows a subset of keyframes and landmarks to be optimised at any one time. A similar representation is also adopted by Lim *et al.* [Lim *et al.*, 2014], allowing their system to perform local bundle adjustment on a small subset of keyframes.

There are also systems which use a hybrid representation. For example, Lim *et al.* [Lim *et al.*, 2011] proposed a SLAM system which alternates bundle adjustment in a local window with global optimisation of keyframe segments. The bundle adjustment in a local window is done using a relative parameterisation whereas the global optimisation of keyframe segments is done using a global parameterisation. A similar representation is proposed by Strasdat *et al.*'s [Strasdat *et al.*, 2011] double window optimisation (DWO) system. In DWO, an inner window performs local bundle adjustment [Sibley *et al.*, 2009] on a small subset of keyframes whereas the outer window marginalises landmarks to perform pose-graph optimisation.

## 3 Incremental Structure from Motion

This section describes the point tracking strategy and the graph optimisation technique used in the SFM pipeline, independent from the types of parameterisations to be used.

### 3.1 Visual Tracking

The first step in the SFM pipeline is calculation of the initial camera pose from the incoming images. In this paper, we use the front-end of the monocular visual SLAM system described in [Lui and Drummond, 2015]. Note that all the measurements are in relative coordinates, although the state representation can be either in local or global coordinate frame. Here, we provide a brief summary of the visual tracking component. The visual tracker computes the pose of the current camera relative to an existing keyframe in the graph using ORB feature

matches [Rublee *et al.*, 2011] as well as the landmarks that are observed by the current camera. In order to get better accuracy, four existing images are selected to provide 2D measurements to the tracker.

The image with the highest number of feature matches with the current view provides the local coordinate system for computing the current measurement. We call this camera the *master* view. Among the images that are connected to the master view through an edge in the global graph, three complementary views that provide the highest number of feature matches to the current view are selected, and are denoted as *neighbour* views, see Fig. 1. Using these 2D feature matches, the visual tracker minimises the re-projection error of the landmarks from the current view into the four selected cameras from the graph, and at the same time estimates the inverse distance of each landmark. Subsequently, as the camera moves around, the master view, and hence the local coordinate frame is passed from one keyframe to another. In order to improve the efficiency of feature matching, we restrict feature matching to be performed between the current view and the cameras that are in the local neighbourhood of the previous master view and its adjacent vertices. For more details on the visual tracking system, we refer the reader to [Lui and Drummond, 2015].

### 3.2 Graph Optimization

Assuming Gaussian noise models, we formulate the SFM problem as an incremental nonlinear least squares (NLS) over a set of variables  $\theta = [\theta_1 \dots \theta_n]$ , the camera poses given by the set  $\mathbf{c} = [c_1 \dots c_{nc}]$  and the points on the structure given by the set  $\mathbf{p} = [p_1 \dots p_{np}]$ , together forming the state  $\theta = [\mathbf{c}, \mathbf{p}]$ . We want to find the optimal configuration satisfying a set of measurements,  $\mathbf{z} = [z_1 \dots z_m]$ , given by the reprojected points on the image. This can be done by performing a maximum likelihood estimation (MLE):

$$\theta^* = \operatorname{argmax}_{\theta} P(\theta | \mathbf{z}) = \operatorname{argmin}_{\theta} \{-\log(P(\theta | \mathbf{z}))\}, \quad (1)$$

where  $Pr(\cdot)$  is the projection function of a point,  $p_j$ , onto the camera,  $c_i$ . Each observation is assumed with zero-mean Gaussian noise with the covariance  $\Sigma_k$  and we measure the re-projection error:

$$P(z_k | c_i, p_j) \propto \exp\left(-\frac{1}{2} \|z_k - Pr_k(c_i, p_j)\|_{\Sigma_k}^2\right), \quad (2)$$

where  $z_k$  is the actual pixel measurement. Note that this paper considers only re-projection error minimisation, other measurement functions are not analysed.

Finding the MLE from (1) is done by solving the fol-

lowing nonlinear least squares problem:

$$\theta^* = \operatorname{argmin}_{\theta} \left\{ \frac{1}{2} \sum_{k=1}^m \|z_k - Pr_k(c_i, p_j)\|_{\Sigma_k}^2 \right\}. \quad (3)$$

Iterative methods such as Gauss-Newton (GN) or Levenberg-Marquard (LM) are used to find the solution of the NLS in (3). An iterative solver starts with an initial point  $\theta^0$  and, at each step, computes a correction  $\delta$  towards the solution. For small  $\|\delta\|$ , a Taylor series expansion leads to linear approximations in the neighbourhood of  $\theta^0$ :

$$\tilde{\mathbf{e}}(\theta^0 + \delta) \approx \mathbf{e}(\theta^0) + J\delta, \quad (4)$$

where  $\mathbf{e} = [e_1, \dots, e_m]^T$  is the set of all nonlinear reprojection errors between the observed and reprojected 3D points,  $e_k(c_i, p_j, z_k) = z_k - Pr_k(c_i, p_j)$ , with  $[c_i, p_j] \subseteq \theta$  and  $J$  is the Jacobian matrix which gathers the derivative of the components of  $\mathbf{e}$ .

Thus, at each  $i^{\text{th}}$  iteration, a linear LS problem is solved:

$$\delta^* = \operatorname{argmin}_{\delta} \frac{1}{2} \|A\delta - \mathbf{b}\|^2, \quad (5)$$

where the  $A = \Sigma^{-T} \backslash^2 J(\theta^i)$  is the system matrix,  $\mathbf{b} = \Sigma^{-T} \backslash^2 \mathbf{e}(\theta^i)$  the right hand side (r.h.s.) and  $\delta = (\theta - \theta^i)$  the correction to be calculated [Dellaert and Kaess, 2006]. The the minimum is attained where the first derivative cancels:

$$A^T A \delta = A^T \mathbf{b} \quad \text{or} \quad \Lambda \delta = \boldsymbol{\eta}, \quad (6)$$

with  $\Lambda = A^T A$ , the square symmetric system matrix and  $\boldsymbol{\eta} = A^T \mathbf{b}$ , the right hand side. The solution to the linear system can be obtained either by sparse matrix factorization followed by backsubstitution or by linear iterative methods. After computing  $\delta$ , the new linearization point becomes  $\theta^{i+1} = \theta^i \oplus \delta$ .

In BA applications the initial solution  $\theta^0$  can be relatively far from the minimum, therefore LM is preferred over the GN methods. LM is based on efficient damping strategies which allow convergence even from poor initial solutions. For that, LM solve a slightly modified variant of (6), which involves a damping factor  $\lambda$ :

$$(\Lambda + \lambda \bar{D})\delta = \boldsymbol{\eta} \quad \text{or} \quad H\delta = \boldsymbol{\eta}, \quad (7)$$

where  $\bar{D}$  can be either the identity matrix,  $\bar{D} = I$ , or the diagonal of the matrix  $\Lambda$ ,  $\bar{D} = \operatorname{diag}(\Lambda)$ .

### 3.3 Incremental SFM

In an online application the system grows every step. New measurements (2) are added to the system incrementally. A new solution  $\theta$  is computed only when all the measurements corresponding to the current camera

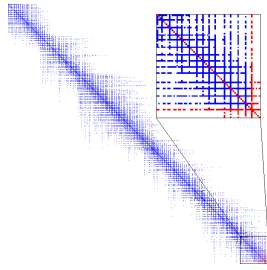


Figure 2: System matrix of the New College dataset, after a new camera and the associated landmarks have been added. Updated values encoded in red.

pose are integrated into the system. In general, given that in robotic applications most of the points are seen only by a handful of cameras, the updates have relatively small rank. An example from of that can be seen in Fig. 2. The size depends only on the number of cameras seeing a point and those cameras are close to each other. Therefore, online SFM can highly benefit from the incremental solving methods introduced in [Polok *et al.*, 2013]. At the same time, by checking the magnitude of the correction  $\delta$  corresponding to each variable at each nonlinear iteration, a partial, fluid re-linearization of the system can be obtain similar to the one proposed in [Kaess *et al.*, 2011], which in turn translates to efficiency.

Another factor which affects the computational complexity is the sparsity of the system matrix  $\Lambda$ . In SFM the sparsity is affected by the number of cameras seeing each point as well as how the points are parameterised in the system. The sparsity of the system matrix is directly related to the connectivity in the underlying graph. Figure 4 shows the graph structure of the SFM problem. We use factor graphs graphical models to represent the joint distribution in (1). A desired parameterisation is the one that reduces the graph connectivity maintaining a good accuracy of the estimate.

## 4 Parameterisations in SFM

As mentioned in Section 2, there are several approaches to parametrising the SFM problem. This section describes the parameterisations examined in detail in this paper.

### 4.1 Point Parameterisations

Frequently, landmarks in the environment are parameterised as 3D points in *Euclidean coordinates*,  $\mathbf{p}_i = [x_j, y_j, z_j]^\top$  resulting in a 3D vector. Nevertheless, this parameterisation allows only estimation up to scale. Scale can be introduced as a parameter resulting in a 4D vector of homogenous coordinates  $\mathbf{p}_j = q_j[x_j, y_j, z_j, 1]^\top$ , where  $q$  is the scale factor.

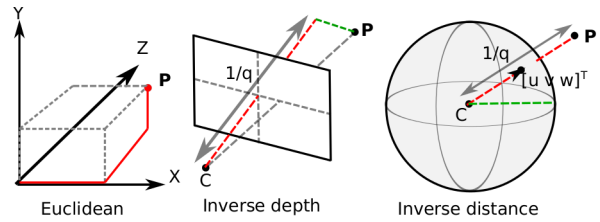


Figure 3: An illustration of the different landmark parameterisations in SFM. Left: Euclidean, where the scale factor  $q$  is 1.0. Center: Inverse depth, where  $1/q$  represents depth and lies in the direction of the camera’s principal axis. Right: Inverse distance, where  $1/q$  points towards the direction of the landmark from the camera center.

A commonly used factor is  $q_j = 1/z_j$ , which yields the *inverse depth* parameterisation of the points, resulting in a 3D vector  $\mathbf{p}_j = [x_j/z_j, y_j/z_j, 1/z_j]^\top$ . If the scale is  $q_j = 1/\sqrt{x_j^2 + y_j^2 + z_j^2}$ , the landmark  $\mathbf{p}_j$  becomes:

$$\mathbf{p}_j = \frac{[x_j, y_j, z_j]^\top}{\sqrt{x_j^2 + y_j^2 + z_j^2}}. \quad (8)$$

The landmark  $\mathbf{p}_j$  in (8) can be re-written in a more compact form  $\mathbf{p}_j = [u_j, v_j, w_j, q_j]^\top$ , where  $[u_j, v_j, w_j]^\top$  is a unit vector representing the normalised camera coordinates of a 2D measurement. The scale factor  $q_j$ , on the other hand, represents the *inverse distance* of the landmark from the observing camera. In this parameterisation, the only variable parameter is the inverse distance  $q$ , while the normalised camera coordinates remain constant. Figure 3 illustrates the difference between the three different landmark parameterisations in SFM.

### 4.2 Camera Parameterisations

If the robot equipped with a camera can freely move in the 3D space, camera poses can be parameterised using 6D vectors. It is common to consider a camera pose as an element of the Lie algebra  $\hat{c}_i \in \mathfrak{se}(3)$  of the special Euclidean group  $SE(3)$  with  $\hat{c}_i$  being the matrix form of the pose  $c_i = [v, \omega]^\top$ ,  $c_i \in \mathbb{R}^6$ :

$$\hat{c}_i = \begin{bmatrix} [\omega]_\times & v \\ 0_{1 \times 3} & 0 \end{bmatrix}, \quad (9)$$

with  $[\omega]_\times$  the skew symmetric matrix of the rotation component  $\omega \in \mathbb{R}^3$ , and  $v \in \mathbb{R}^3$  the translation component. In this case, the nonlinear least squares problem in (3) can be solved performing iterative optimisations on the manifold.

Optimising only for the camera poses requires the camera parameters to be known so that the re-projection error in (4) can be calculated. In general, in an SFM

robotic application those parameters can be estimated a-priori by calibrating the camera. On the other hand, performing an on-line structure from motion using  $SE(3)$  poses is problematic, due to the scale drift. In case of monocular observation of a forward motion, the scale is ambiguous. Therefore, a good robust estimate of the scale is required in order to obtain a consistent solution. For instance detection of the ground plane and calculation of the scale as a function of distance of a camera from the ground (assuming that the camera does not move vertically) can be used.

Nevertheless, the scale can be better estimated during the optimisation process, and this can be achieved by considering the camera poses as elements of the Lie algebra  $\hat{c}_i \in \mathfrak{sim}(3)$  of the Similarity group  $Sim(3)$  with:

$$\hat{c}_i = \begin{bmatrix} [\omega]_{\times} + qI_{3 \times 3} & v \\ 0 & 0 \end{bmatrix}, \quad (10)$$

$q \in \mathbb{R}$  being the scale factor [Strasdat *et al.*, 2011].

### 4.3 Point Coordinate Frames

Every point  $\mathbf{p}_j$  from the state vector  $\theta$  is expressed in a reference frame, either globally in the world coordinate frame or locally in one of the camera's coordinate frame.

#### Points in Absolute Coordinate Frames

It is common to consider that the SFM problem has a common world reference frame for all points and cameras,  $\{\mathcal{W}\}$ . In this case the measurement function  $Pr_k(c_i, p_j)$  projects a point  ${}^{\mathcal{W}}p_j$  from the world coordinates to image pixels in the image coordinates  ${}^{\mathcal{I}}p_j$ . If  ${}^{\mathcal{C}}p_j$  is the point in the camera coordinate frame  $\{\mathcal{C}\}$  and  $K_i$  are the intrinsic parameters of the camera  $c_i$  in a matrix form, we can write this projection as:

$$T_i = \exp({}^{\mathcal{W}}\hat{c}_i), \quad (11)$$

$${}^{\mathcal{C}}p_j = T_i^{-1} {}^{\mathcal{W}}p_j, \quad (12)$$

$${}^{\mathcal{I}}p_j = K_i {}^{\mathcal{C}}p_j, \quad (13)$$

where  $T_i$  is the transformation matrix corresponding to the pose of camera  $c_i$  (in world coordinate frame),  $\exp(\cdot)$  being the exponential map from the Lie algebra to the manifold for any camera parameterisation described in section 4.2. Note that in (13), the coordinates of  ${}^{\mathcal{I}}p_j$  are homogenous, but the measurement function  $Pr_k(c_i, p_j)$  returns them dehomogenized. The factor graph representation of the global parameterisation is shown in Fig. 4 a), each binary factor being a function  $\phi({}^{\mathcal{W}}c_i, {}^{\mathcal{W}}p_j)$  and represented using light-blue points.

#### Points in Local Coordinate Frames

In a relative representation, each point is represented in the coordinate frame of a selected camera. We will refer to this camera as the *owner* camera. The choice

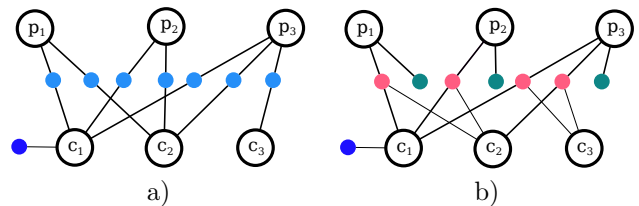


Figure 4: Factor graph representation of the SFM problem. Points are represented in a) global coordinates, measurements are binary factors (blue), or b) local coordinates, measurements are unary factors (green) and ternary factors (red). Dark blue is the prior on the first camera pose.

of the owner camera has to be made when the point is added to the system, which is typically immediately after being triangulated. If a point is triangulated e.g. from two cameras, one of them must become its owner. The second camera is preferable, as it is likely closer to any other cameras that might re-observe the point in the future, making the transformations more local.

The easiest way to understand the local coordinate frame representation is by looking at the associated factor graph. Fig. 4 b) shows the types of factors involved in a local coordinates representation. Each point  $p_j$  can be "observed" by a camera  $c_o$  but actually "represented" in the coordinate frame  $\{\mathcal{R}\}$  of the owner camera  $c_r$ .

A unary factor corresponds to the owner camera  $c_r$  observing the point  $p_j$  in its own coordinate frame. The measurement function  $Pr_k({}^{\mathcal{W}}c_r, {}^{\mathcal{R}}p_j)$  is:

$${}^{\mathcal{I}}p_j = K_r {}^{\mathcal{R}}p_j. \quad (14)$$

Note that the pose of  $c_r$  does not play a role in this function, hence the corresponding factor is unary. On the other hand, a ternary factor  $\phi({}^{\mathcal{W}}c_o, {}^{\mathcal{W}}c_r, {}^{\mathcal{R}}p_j)$  is needed to re-project  ${}^{\mathcal{R}}p_j$  into  $c_o$ :

$${}^{\mathcal{W}}p_j = T_r {}^{\mathcal{R}}p_j, \quad (15)$$

$${}^{\mathcal{O}}p_j = T_o^{-1} {}^{\mathcal{W}}p_j, \quad (16)$$

$${}^{\mathcal{I}}p_j = K_o {}^{\mathcal{O}}p_j, \quad (17)$$

$T_r = \exp(\hat{c}_r)$  being the transformation from the owner camera  $c_r$  coordinate frame to the world coordinate frame and subsequently  $T_o^{-1} = \exp(-\hat{c}_o)$  being the transformation that brings it to the observing camera coordinate frame  $\{\mathcal{O}\}$ .

### 4.4 Camera Coordinate Frames

Much like the points, the cameras can also be represented relative to each other. While this may be beneficial from the point of view of convergence, it also changes the structure of the graph substantially. While relative point representation requires ternary factors, relative cameras

with relative points would require arbitrary size factors. Each loop closure would then form a clique in the factor graph, leading to pathological performance in online solving. Therefore, only the case of global camera coordinates is considered in this paper.

## 5 Analysis of Different Parameterisations for Online SFM

In this section we evaluate the accuracy and efficiency of several SFM point parameterisations in online applications. Several types of metrics are proposed in the literature, mainly testing the accuracy of the estimated camera poses not the reconstruction itself. If a ground truth is available, relative pose error (RPE) can be calculated. It was used in [Kummerle *et al.*, 2009] and [Sturm *et al.*, 2012] for the evaluation of the graph SLAM and Kinect SLAM, and involves calculating the difference between the estimated relative poses and the true relative poses of the camera. A more intuitive way is to compare the absolute trajectory error (ATE) by registering the ground truth and the estimated camera pose graph [Sturm *et al.*, 2012].

Table 1 evaluates the precision of two structure from motion datasets using the above-mentioned metrics. The Loop dataset is a synthetic dataset, generated for evaluation of the frontend. It consists of a sequence of 180 poses, observing uniformly distributed points while moving in a circle. The last cameras re-observe the first points, i.e. the loop is closed. The second dataset is the TUM Frei 3 dataset, which is a sequence of monocular images with ground truth generated by a motion capture system. The estimated poses are aligned to the ground truth poses using the Kabsch algorithm and then the error metrics are evaluated. The precisions of different parameterisations are comparable, with the notable exception of global inverse distance. This parameterisation is somewhat limited, as the landmarks are restricted to a single degree of freedom (they are only free to move along the axis of projection) and may not converge well.

Table 2 evaluates of the same SFM datasets, with the addition of the New College dataset for which the ground truth is not available. Given that the precision of the different parameterisations is similar, this second test aims to show their influence on online optimisation algorithm, analysing the convergence and the efficiency. The re-projection error is evaluated as  $\chi^2$ , mean of the re-projection error in pixels and RMSE of the same. In addition, we calculated the condition numbers of the information matrix for each dataset and parameterisation, and also measured the time spent in the nonlinear least squares solver. The *local parameterisations* have consistently better condition number, which is important in solving large systems, or in cases where precise floating point calculations are restricted, such as on a GPU.

The inverse distance parameterisations are faster (while the global inverse distance is slightly faster than local, possibly due to simpler factors), although at the cost of slightly lower precision.

With the incremental solving on mind, the incremental updates were analysed and compared. Figure 5 plots the incremental updates on the New College dataset using the *global Euclidean parameterisation*. Note that the size of the state is growing almost linearly, as every camera observes new landmarks. The value of  $norm(\delta)$  is changing in the whole time span, which indicates that the solution is being optimized, rather than settled. The important part to note are the numbers of changing variables. To calculate these numbers, norm of the segment of  $\delta$  corresponding to each variable was calculated and compared to the thresholds ( $10^{-6}$ ,  $10^{-5}$  and  $10^{-4}$ ). We can see that all the variables change, all the time, and incremental solving is therefore bound to run in exponential time.

On the other hand, in Fig. 6, there is the same plot for the *local inverse depth parameterisation*. Note that this time, the number of changing variables does not follow the number of variables in the system, and with time settles around 9000 variables being updated at every frame for the  $10^{-6}$  threshold, or about 4000 for  $10^{-5}$ . Finally, in Fig. 7, shows a comparison of update sizes of different parameterisations for the threshold  $10^{-5}$ . Note that the updates of the inverse distance have a different dimensionality than Euclidean or inverse depth, and are plotted on the secondary axis. We can see that both the local Euclidean and local inverse depth parameterisation settle with constant number of updates per frame. Notably, the local inverse distance does not hold this trend, and despite the drop around the 70th frame, keeps rising.

In Table 3, there are results of evaluations on bundle adjustment datasets. While not quantitatively the same as the results on the SfM datasets, the parameterisations examined in this paper apply to bundle adjustment solving as well. The datasets include the Guildford Cathedral<sup>1</sup> and Fast & Furious 6 which was kindly provided by Double Negative Visual Effects<sup>2</sup>. Similar behaviour to the SFM datasets can be observed in here as well.

## 6 Conclusions and Future Work

In this paper, we have examined some of the popular parameterisations used in the structure from motion framework, with focus towards globally consistent incremental solving. The current state of the art incremental SFM solvers aim to work in constant time, in order to provide the ability to map large environments. Nevertheless, the commonly used techniques for that are ei-

<sup>1</sup>can be obtained at <http://cvssp.org/impart/>

<sup>2</sup><http://www.dneg.com/>

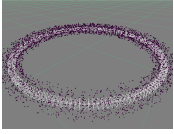
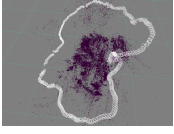
| Dataset   | Param.           | Error                         |                              |                              |
|---|------------------|-------------------------------|------------------------------|------------------------------|
|   |                  | RMSE ATE                      | RMSE RPE                     | RMSE RPE all-all             |
| <br>Loop | loc. Euclidean   | 0.000182 m, 0.020595°         | 0.000132 m, 0.024064°        | 0.000358 m, 0.020469°        |
|   | loc. inv. depth  | 0.000186 m, 0.020732°         | 0.000132 m, 0.024109°        | 0.000363 m, 0.020619°        |
|   | loc. inv. dist.  | <b>0.000109 m, 0.014099°</b>  | <b>0.000116 m, 0.019999°</b> | <b>0.000247 m, 0.013945°</b> |
|   | glob. Euclidean  | 0.000405 m, 0.021853°         | 0.000134 m, 0.024093°        | 0.000531 m, 0.021737°        |
|   | glob. inv. depth | 0.001870 m, 0.072085°         | 0.000564 m, 0.033105°        | 0.002158 m, 0.070572°        |
|   | glob. inv. dist. | 0.005204 m, 0.425437°         | 0.002362 m, 0.283630°        | 0.008588 m, 0.423213°        |
| <br>TUM  | loc. Euclidean   | 0.622578 m, 10.620843°        | 0.022810 m, 0.626021°        | 0.745422 m, 9.734252°        |
|   | loc. inv. depth  | 0.645346 m, 11.304622°        | 0.035738 m, 1.231776°        | 0.762337 m, 10.703952°       |
|   | loc. inv. dist.  | 0.646779 m, 11.303658°        | 0.034766 m, 1.178380°        | 0.761142 m, 10.692832°       |
|   | glob. Euclidean  | <b>0.619225 m, 10.618006°</b> | <b>0.022909 m, 0.659206°</b> | <b>0.742262 m, 9.732721°</b> |
|   | glob. inv. depth | 0.620440 m, 10.676661°        | 0.023971 m, 0.684937°        | 0.743851 m, 9.793333°        |
|   | glob. inv. dist. | 0.623242 m, 10.624283°        | 0.027379 m, 0.899663°        | 0.745059 m, 9.783204°        |

Table 1: Error evaluation.

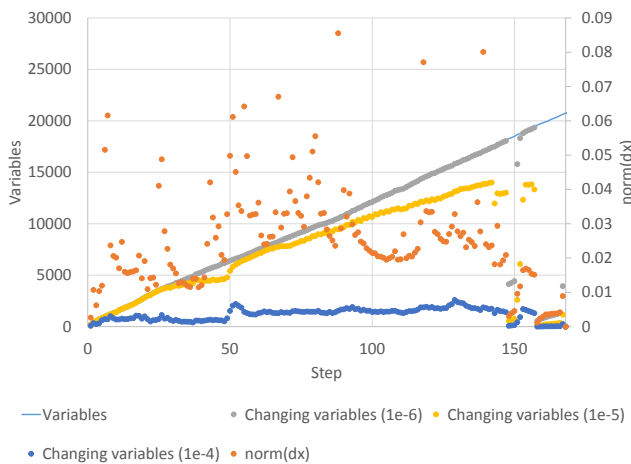


Figure 5: Comparison of effects of the global Euclidean parameterization on the incremental updates on the New College dataset.

ther postponing the global optimization (PTAM), use an iterative solver rather than a direct one [Eustice *et al.*, 2006] or produce only locally consistent solution by the means of windowed optimization. In this paper, we show that efficient and scalable globally consistent SFM is possible with the right parameterisation. Using *local inverse depth parameterisation* the number of variables affected by integrating new measurements into the optimiser stays bounded, even if the size of the state continuously grows. In our future work, we will focus on implementation of an incremental NLS solver, suitable for optimising such problems in constant time.

## 7 Acknowledgements

We are extremely grateful to the ARC Centre of Excellence for Robotic Vision, project number CE140100016

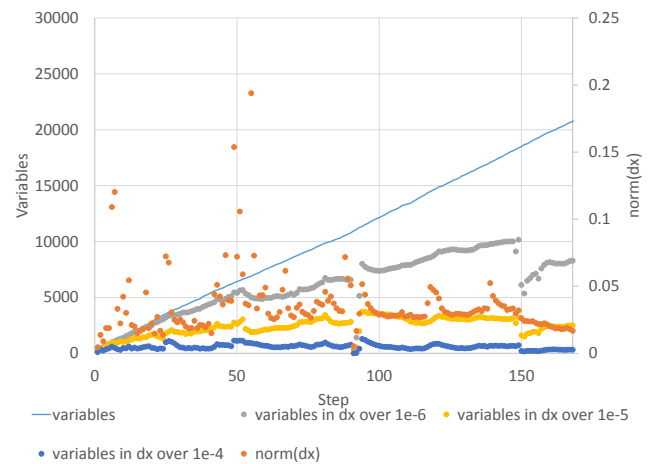


Figure 6: Comparison of effects of the local inverse depth parameterization on the incremental updates on the New College dataset.

for funding this research. We acknowledge the support of NVIDIA Corporation with the donation of the GPUs used for this research. The research leading to these results has received funding from the European Union, 7<sup>th</sup> Framework Programme grants 316564-IMPART and the IT4Innovations Centre of Excellence, grant n. CZ.1.05/1.1.00/02.0070, supported by Operational Programme Research and Development for Innovations funded by Structural Funds of the European Union and the state budget of the Czech Republic.

## References

[Agarwal *et al.*, 2010] Sameer Agarwal, Noah Snavely, Steven M. Seitz, and Richard Szeliski. Bundle adjustment in the large. In *Computer Vision ECCV 2010*, pages 29–42. 2010.

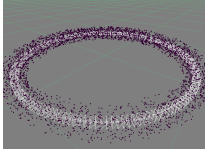
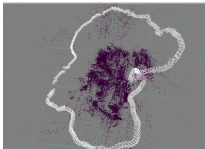
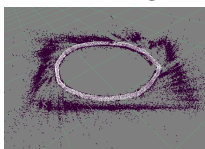
| Dataset  | Parameterization | Error             |                |                | Cond. nr.                              | Time [sec]       |
|--|------------------|-------------------|----------------|----------------|--|------------------|
|  |                  | $\chi^2$          | Mean [px]      | RMSE [px]      |  |                  |
| <br>Loop        | loc. Euclidean   | <b>75561.98</b>   | <b>0.63917</b> | <b>0.79471</b> | $1.07 \cdot 10^{16}$                   | 21.265321        |
|  | loc. inv. depth  | 75590.73          | 0.63925        | 0.79486        | <b><math>3.57 \cdot 10^{15}</math></b> | 21.375869        |
|  | loc. inv. dist.  | 77992.62          | 0.64475        | 0.80739        | $3.38 \cdot 10^{15}$                   | 13.057721        |
|  | glob. Euclidean  | 75581.48          | 0.63926        | 0.79482        | $6.60 \cdot 10^{14}$                   | 20.821423        |
|  | glob. inv. depth | 163359.53         | 0.75338        | 1.16851        | $6.32 \cdot 10^{14}$                   | 20.875177        |
|  | glob. inv. dist. | 3469284.72        | 3.55021        | 5.38491        | $1.73 \cdot 10^{15}$                   | <b>12.867192</b> |
| <br>TUM         | loc. Euclidean   | 16350991.75       | 2.07952        | 8.47533        | $5.17 \cdot 10^{21}$                   | 56.630716        |
|  | loc. inv. depth  | <b>5456575.52</b> | <b>1.52743</b> | <b>4.99786</b> | <b><math>1.31 \cdot 10^{15}</math></b> | 56.541088        |
|  | loc. inv. dist.  | 7836349.46        | 1.61037        | 5.88475        | $1.35 \cdot 10^{15}$                   | 50.589076        |
|  | glob. Euclidean  | 14212345.55       | 2.16143        | 7.90505        | $4.97 \cdot 10^{23}$                   | 57.095413        |
|  | glob. inv. depth | 16679822.96       | 2.18666        | 8.53513        | $2.04 \cdot 10^{24}$                   | 56.812929        |
|  | glob. inv. dist. | 14090046.92       | 2.50113        | 7.83685        | $1.17 \cdot 10^{23}$                   | <b>50.116903</b> |
| <br>new-college | loc. Euclidean   | 1420827.89        | 0.93290        | 3.23703        | $1.31 \cdot 10^{28}$                   | 16.780468        |
|  | loc. inv. depth  | <b>676521.75</b>  | <b>0.79121</b> | <b>2.31696</b> | <b><math>3.64 \cdot 10^{18}</math></b> | 16.814846        |
|  | loc. inv. dist.  | 1006022.72        | 0.84702        | 2.74105        | $6.86 \cdot 10^{18}$                   | 14.321912        |
|  | glob. Euclidean  | 834114.98         | 0.78745        | 2.53967        | $2.74 \cdot 10^{30}$                   | 16.499947        |
|  | glob. inv. depth | 856986.80         | 0.82171        | 2.56488        | $2.56 \cdot 10^{29}$                   | 16.440494        |
|  | glob. inv. dist. | 1253513.47        | 1.03306        | 3.04523        | $1.67 \cdot 10^{31}$                   | <b>13.440248</b> |

Table 2: Error evaluation on SFM datasets. The best values for each dataset and category are set in bold.

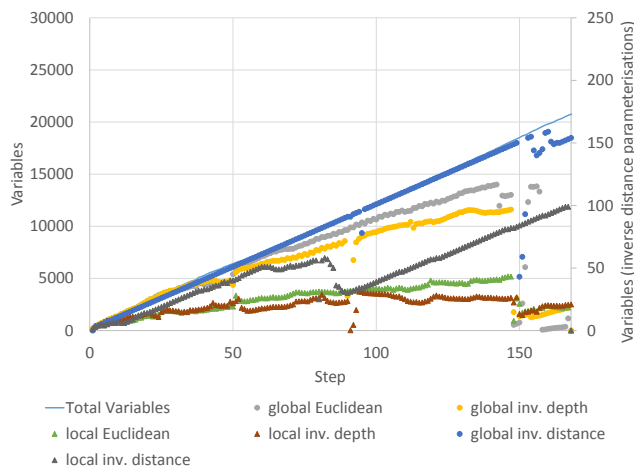


Figure 7: Comparison of effects of the different parameterizations on the size of the incremental updates on the New College dataset.

- [Bosse *et al.*, 2003] M. Bosse, Paul Newman, John Leonard, M. Soika, W. Feiten, and S. Teller. An ATLAS framework for scalable mapping. In *IEEE International Conference on Robotics and Automation*, 2003.
- [Dellaert and Kaess, 2006] F. Dellaert and M. Kaess. Square Root SAM: Simultaneous localization and mapping via square root information smoothing. *Intl. J. of Robotics Research*, 25(12):1181–1203, Dec 2006.
- [Eade and Drummond, 2006] E. Eade and T. Drum-

mond. Scalable Monocular SLAM. In *Computer Vision and Pattern Recognition*, volume 1, pages 469–476. IEEE, 2006.

- [Eade and Drummond, 2008] Ethan Eade and Tom Drummond. Unified loop closing and recovery for real time monocular SLAM. In *British Machine Vision Conference*, 2008.
- [Eustice *et al.*, 2006] R.M. Eustice, H. Singh, J.J. Leonard, and M.R. Walter. Visually mapping the RMS Titanic: Conservative covariance estimates for SLAM information filters. *Intl. J. of Robotics Research*, 25(12):1223–1242, Dec 2006.
- [Guivant and Nebot, 2001] J. Guivant and E. Nebot. Optimization of simultaneous localization and map building algorithm for real time implementation. *IEEE Transactions on Robotics and Automation*, 17:242–257, 2001.
- [Indelman and Dellaert, 2015] V. Indelman and F. Dellaert. Incremental light bundle adjustment: Probabilistic analysis and application to robotic navigation. In *New Development in Robot Vision*, volume 23, pages 111–136. Springer Berlin Heidelberg, 2015.
- [Kaess *et al.*, 2011] M. Kaess, H. Johannsson, R. Roberts, V. Ila, J. Leonard, and F. Dellaert. iSAM2: Incremental smoothing and mapping with fluid relinearization and incremental variable reordering. In *IEEE Intl. Conf. on Robotics and Automation (ICRA)*, Shanghai, China, May 2011.



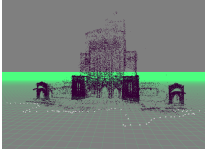
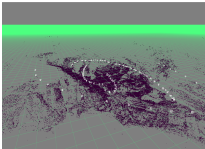
| Dataset   | Parameterization | Error             |                |                | Cond. nr.                               | Time [sec]      |
|---|------------------|-------------------|----------------|----------------|---|-----------------|
|   |                  | $\chi^2$          | Mean [px]      | RMSE [px]      |   |                 |
| <br>Cathedral        | loc. Euclidean   | 3062204.52        | 2.19059        | 2.69325        | $1.26 \cdot 10^{15}$                    | 5.588799        |
|   | loc. inv. depth  | <b>2977150.96</b> | <b>2.14974</b> | <b>2.65559</b> | <b><math>8.57 \cdot 10^{14}</math></b>  | 5.684552        |
|   | loc. inv. dist.  | 24672548.94       | 5.44801        | 7.64481        | $1.10 \cdot 10^{15}$                    | 3.714308        |
|   | glob. Euclidean  | 2999076.64        | 2.16071        | 2.66535        | $1.04 \cdot 10^{15}$                    | 5.286748        |
|   | glob. inv. depth | 3235493.29        | 2.26074        | 2.76841        | $2.71 \cdot 10^{15}$                    | 5.304403        |
|   | glob. inv. dist. | 5651805.98        | 2.91999        | 3.65893        | $2.02 \cdot 10^{16}$                    | <b>3.587127</b> |
| <br>Fast & Furious 6 | loc. Euclidean   | 832154.12         | 1.06404        | 1.33594        | <b><math>1.841 \cdot 10^{15}</math></b> | 4.279088        |
|   | loc. inv. depth  | 803669.41         | 1.03367        | 1.31288        | $3.02 \cdot 10^{15}$                    | 4.216713        |
|   | loc. inv. dist.  | 8076903.30        | 2.53310        | 4.16205        | $3.29 \cdot 10^{15}$                    | 3.150134        |
|   | glob. Euclidean  | <b>802978.28</b>  | <b>1.03320</b> | <b>1.31231</b> | $1.844 \cdot 10^{15}$                   | 3.954314        |
|   | glob. inv. depth | 998813.47         | 1.17967        | 1.46362        | $1.02 \cdot 10^{16}$                    | 3.866687        |
|   | glob. inv. dist. | 1398193.67        | 1.41222        | 1.73168        | $8.78 \cdot 10^{15}$                    | <b>2.983877</b> |

Table 3: Error evaluation on bundle adjustment datasets. The best values for each dataset and category are set in bold.

- [Klein and Murray, 2007] Georg Klein and David Murray. Parallel Tracking and Mapping for Small AR Workspaces. In *IEEE and ACM International Symposium on Mixed and Augmented Reality*, pages 1–10. Ieee, November 2007.
- [Konolige and Agrawal, 2008] Kurt Konolige and Motilal Agrawal. FrameSLAM : from Bundle Adjustment to Realtime Visual Mapping. *IEEE Transactions on Robotics*, 24(5):1–11, 2008.
- [Kummerle *et al.*, 2009] Rainer Kummerle, Bastian Steder, Christian Dornhege, Michael Ruhnke, Giorgio Grisetti, Cyrill Stachniss, and Alexander Kleiner. On measuring the accuracy of slam algorithms. *Autonomous Robots*, 27(4), 2009.
- [Lim *et al.*, 2011] Jongwoo Lim, Jan-Michael Frahm, and Marc Pollefeys. Online environment mapping. In *Computer Vision and Pattern Recognition*, pages 3489–3496. IEEE, June 2011.
- [Lim *et al.*, 2014] Hyon Lim, Jongwoo Lim, and H Jin Kim. Real-Time 6-DOF Monocular Visual SLAM in a Large-Scale Environment. In *International Conference on Robotics and Automation (ICRA)*, pages 1532–1539, 2014.
- [Lui and Drummond, 2015] Vincent Lui and Tom Drummond. Image based optimisation without global consistency for constant time monocular visual slam. In *IEEE International Conference on Robotics and Automation*, pages 5799–5806, 2015.
- [Mei *et al.*, 2010] Christopher Mei, Gabe Sibley, Mark Cummins, Paul Newman, and Ian Reid. RSLAM: A System for Large-Scale Mapping in Constant-Time Using Stereo. *International Journal of Computer Vision*, 94(2):198–214, June 2010.
- [Newcombe and Davison, 2010] Richard A. Newcombe and Andrew J. Davison. Live dense reconstruction with a single moving camera. In *IEEE Conference on Computer Vision and pattern Recognition*, 2010.
- [Pollefeys *et al.*, 2001] Marc Pollefeys, Maarten Vergauwen, Kurt Cornelis, Jan Tops, Frank Verbiest, and Luc Van Gool. Structure and motion from image sequences. In *Proc. Conf. on Optical 3-D Measurement Techniques*, pages 251–258, 2001.
- [Polok *et al.*, 2013] Lukas Polok, Viorela Ila, Marek Solony, Pavel Smrz, and Pavel Zemcik. Incremental block cholesky factorization for nonlinear least squares in robotics. In *Proceedings of the Robotics: Science and Systems 2013*, 2013.
- [Rublee *et al.*, 2011] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to SIFT or SURF. In *International Conference on Computer Vision (ICCV)*, pages 2564–2571. IEEE, November 2011.
- [Sibley *et al.*, 2009] Gabe Sibley, Christopher Mei, Ian Reid, and Paul Newman. Adaptive Relative Bundle Adjustment. In *Robotics: Science and Systems*, pages 1–8, 2009.
- [Strasdat *et al.*, 2011] Hauke Strasdat, Andrew J. Davison, J. M. M. Montiel, and Kurt Konolige. Double Window Optimisation for Constant Time Visual SLAM. In *International Conference on Computer Vision*, pages 2352–2359, 2011.
- [Sturm *et al.*, 2012] J. Sturm, N. Engelhard, F. Endres, W. Burgard, and D. Cremers. A benchmark for the evaluation of rgb-d slam systems. In *Proc. of the International Conference on Intelligent Robot Systems (IROS)*, Oct. 2012.