Monocular Panoramic 3D Reconstruction Based on a Particle Filter

Frank Pagel
Fraunhofer Institute Optronics, System Technologies and Image Exploitation (IOSB)
Fraunhoferstr.1, 76131 Karlsruhe, Germany

ABSTRACT
This paper addresses the issue of generating a panoramic view and a panoramic depth maps using only a single camera. The proposed approach first estimates the egomotion of the camera. Based on this information, a particle filter approximates the 3D structure of the scene. Hence, 3D scene points are modeled probabilistically. These points are accumulated in a cylindric coordinate system. The probabilistic representation of 3D points is used to handle the problem of visualizing occluding and occluded scene points in a noisy environment to get a stable data visualization. This approach can be easily extended to calibrated multi-camera applications (even with non-overlapping field of views).

Keywords: Reconstruction, Particle Filter, Monocular Vision, Omnidirectional Vision

1. INTRODUCTION
Monocular 3D reconstruction tasks are in the focus of interest for a long period of time. Simultaneous localization and Mapping (SLAM) and structure from motion (SfM) techniques are widely used, e.g. Pagel\textsuperscript{1} or Davison et al.\textsuperscript{2} The goal of this contribution is not only to calculate the structure of the scene based on egomotion information. Even more, a panoramic view of the scene texture as well as a panoramic view of the depth structure is generated. This contribution covers the topics of monocular 3D reconstruction and temporal data fusion for panoramic vision. Furthermore the problem of visualizing noisy point clouds is addressed.

Woock et al.\textsuperscript{3} calculate sparse 3D point clouds with a single camera based on odometry data in real-time. They do not consider the influence of noisy odometry data for the reconstruction process. Woelk et al.\textsuperscript{4} use a 2D particle filter to detect independently moving objects from a moving platform. Their work is inspiring for the following approach.

In this paper, the panoramic view is generated by accumulating 3D points generated by a particle filter. These 3D particles are collected over time in a reference coordinate system and are then projected on a virtual image cylinder. The particle filter decreases computational effort for calculating the 3D point cloud because the 3D coordinates have to be calculated only for a relatively small number of points via triangulation. In fact, the filter distributes the point cloud in the resampling step according to the assumed underlying density function. Another major advantage of the particle filter is that every particle has a weight that indicates the reliability of the 3D point. This circumstance is extremely useful when a large amount of accumulated points have to be handled e.g. in visualization or outlier handling for 3D surface modelling.

Usually, 2D mosaicing techniques (e.g. Szeliski\textsuperscript{5}) seem to fail in arbitrary scenes where no major planar surface is present or the camera is moving forward. A lot of n-view reconstruction techniques have been published over the last years. Pollefeys et al.\textsuperscript{6} generate dense, textured 3D models from multiple views based on motion estimation and the calculation of dense depth maps with stereo techniques. But it is a major concern of this paper to present an algorithm that is capable to work in real-time and hence is able to deliver panoramic views of a highly dynamic and changing scene. Such scenes might be inerity as well as offroad scenarios with an egomotion velocity from approximately 10 to 50 $km/h$. This is also a relevant topic in robotics and for remote-controlled navigation. Because the particle filter delivers no accurate 3D points (rather probabilistic point clouds), it makes no sense to use temporal point referencing algorithms like ICP as is for example used by Fusiello et al.\textsuperscript{7} to built 3D mosaics of an under-water scene. Actually, the referencing is done by transforming the 3D particles in a reference system and accumulating them in a cylindric coordinate system.

The paper is structured as follows: Section 2 presents the basic principle and the implementation details of the particle filter as well as the motion model. In section 3 the cylindric coordinate system is presented that is used for the panoramic visualization. A strategy for handling noisy and/or occluding particles is proposed. Section 4 shows some results.

Further author information:
E-mail: frank.pagel@iosb.fraunhofer.de, Telephone: +49 (0)721 6091 518
2. IMPLEMENTATION OF THE PARTICLE FILTER

2.1 Sequential Monte Carlo Methods

Sequential Monte Carlo (SMC) methods are a set of flexible simulation-based methods for sampling from a sequence of probability distributions. They are often used to solve sequential Bayesian inference problems arising in signal processing. SMC methods approximate a sequence of probability distributions using a large set of random samples, named particles. These particles are propagated over time using prediction, update, measurement and resampling mechanisms. Asymptotically, i.e., as the number of particles goes to infinity, these particle approximations converge towards the sequence of probability distributions. However, for practical implementations, a finite and sometimes quite restricted number of particles has to be considered. It is therefore important to implement efficient sampling strategies in order to sample particles in regions of high probability mass.

In this paper the 3D scene is modeled as a probability distribution. This means that the goal is to determine the probability for a 3D point to lie on a visible, rigid surface in the scene. The particle filter is designed to approximate any arbitrary, non-gaussian density function with respect to the measurements. In the present case, a dynamic number of particles is used to guarantee flexibility and to be able to cover the whole scene as well as scene changes (that is what happens when the camera is moving around). Each particle is scored and these scores are used to approximate the true probability distribution. As a result the set of weighted particles models the static 3D scene.

In this contribution, a particle is considered as a tuple including the state $\theta$, a measurement $z$ and a probability weight $\omega$.

$\theta$ is the position $(X,Y,Z)^T$ in 3D space. The measurement $z$ includes the projected image point $p$ of $P \in \mathbb{R}^3$, the optical flow $\vec{u}(p)$, the egomotion of the camera itself (which is the same for all particles) and the image value $g = I(p + \vec{u})$.

The particle filter processes the following cycle at each time step: Prediction (estimation of the new position of the particles based on the egomotion estimation), update (adding new particles with strong image features and importance samples that are nearly uniformly distributed over the whole image), measurement and scoring (calculating the score for each particle w.r.t. optical flow, egomotion and color measuring) and resampling (approximating the probability distribution w.r.t. the particle scores). This scheme is shown in fig. 1.

![Figure 1. Example scheme of the particle filter process for one time step.](image)

2.2 Motion Model

The motion of a single camera can be described via an Euclidean transformation. The motion of a camera between two time steps is given by a rotation matrix $\Omega(\omega_x, \omega_y, \omega_z) \in \mathbb{R}^{3 \times 3}$ and a translation vector $V = (v_x, v_y, v_z)^T$. It is assumed that the translation vector of one camera’s motion is only given up to a scale factor $s \in \mathbb{R}$ (the motion itself is therefore

*In this paper only gray scale is considered, but in principle color would work, too.
described by a similar transformation). The motion of \( X_{ik} \) (which actually is the inverse motion of the camera) seen with camera \( c_i \) at time \( k \) is given by

\[
X_{ik+1} = \Omega_{ik}X_{ik} + s_iV_{ik}.
\] (1)

Once the motion and the intrinsic camera parameters are known, the 3D point of a corresponding pair of image points in two consecutive frames can be calculated (e. g. Hartley and Zisserman\(^8\)). In this paper the camera is assumed to be calibrated (e. g. Zhang\(^9\)).

2.3 Prediction

At the beginning of a particle filter cycle the state of all particles is predicted according to the motion model. This is a simple Euclidean transformation using the known rotation and translation parameters.

2.4 Update and Importance Samples

At each time step a new set of samples is added to the existing samples. This step is necessary because the particle filter tends to cluster all particles at on 3D location with a high probability. This problem is known as the degeneracy problem. But for the given task it is important to have as much 3D information over the whole image as possible. That is why some additional importance particles are selected in the image plane. They can be considered as anchors or gravitational objects that keep the set of particles in balance. The weight of the importance samples is given by

\[
\gamma = \frac{1}{N_{imp}} \cdot exp \left( - \frac{z_{fx}^2 + z_{fy}^2}{2\sigma_f^2} \right) \in [0, 1/N_{imp}]
\] (2)

where \( N_{imp} \) is the number of importance samples. These weights are usually stronger than the weights of the regular particles because \( N_{imp} \) is relatively small compared to the number of old particles \( N_{old} \), except the projection error is too big and hence the particle has turned out to be a bad choice. The importance samples are chosen to be spread over the whole image as well as lying on significant image structures. Using the egomotion parameters, one can calculate the initial particle position in 3D space by calculating the optical flow and performing a triangulation. Finally, to achieve a consistent representation of the posteriori density, the importance weights become normalized like all the other weights, too. The current number of particles is updated to

\[
N = N_{img} + N_{old}.
\]

2.5 Measurement and Scoring

The likelihood function models the observation process. Here the probability of a 3D scene point lying on a rigid, visible surface is derived.

The 3D point of the particle \( \theta \) is projected into the image plane \( I_t \):

\[
p_t = \pi_t(\theta)
\] (3)

Then the 3D point is transformed using the known egomotion parameters and projected again into the image plane \( I_{t+1} \):

\[
p_{t+1} = \pi_{t+1}(\theta)
\] (4)

So we have two projections, \( p_t \) and \( p_{t+1} \). \( \pi_t(\theta) \) is the function that projects the 3D state point \( \theta \) in the image plane at time \( t \). The optical flow vector \( \vec{u} \) is calculated for \( p_t \) using standard techniques like the tracking algorithm of Tomasi and Kanade.\(^10\) The flow measurement

\[
z_f = p_{t+1} - p_t = (z_{fx}, z_{fy})^T
\] (5)

is assumed to be corrupted by zero mean gaussian distributed noise with variance \( \sigma_f \). As an additional measurement the difference of the image values is used (gray scale or 3 channel color). The color measurement

\[
z_c = I_{t+1}(p_{t+1}) - I_t(p_t)
\] (6)

is also assumed to be corrupted by zero mean gaussian distributed noise with variance \( \sigma_c \). Furthermore, the projection error and the color difference are assumed to be uncorrelated. So the likelihood function for the measurement \( z = (z_{fx}, z_{fy}, z_{c})^T \) is modeled as

\[
p(z|\theta) = \frac{1}{(2\pi)^{3/2}\sigma_f^2\sigma_c^2} \cdot \exp \left( - \frac{1}{2} \cdot \frac{z_{fx}^2}{\sigma_f^2} - \frac{z_{fy}^2}{\sigma_f^2} - \frac{z_{c}^2}{\sigma_c^2} \right).
\] (7)
As the optical flow measurement used in this paper is a feature based approach, features at significant image positions are to prefer. This is reasonable because the optical flow vectors for each projected particle must be calculated. Particles \( \mathbf{p} \) which projections are close to image edges are weighted stronger than features in homogenous regions. The weights for all image positions are given by an edge image \( E_{t+1} \), extracted from \( I_{t+1} \) (e. g. \( I_{t+1} \)), smoothed with a \( m_{\text{smooth}} \times m_{\text{smooth}} \) gaussian kernel \( G_{\text{smooth}} \) with variance \( \sigma^2_{\text{smooth}} \).

It turned out in practice that the degeneracy problem occurs quite often and mostly already after a short tracking period. To countersteer this progress the additional parameter \( \beta \) is introduced, the so called equality factor. The final normalized particle weights are given by

\[
\omega_i := \frac{\left(1 - \beta \right) \cdot p(z_i|\theta_i) + \beta \frac{1}{N}}{\sum_{k=1}^{N} \left(1 - \beta \right) \cdot p(z_k|\theta_k) + \beta \frac{1}{N}} \cdot W_{t+1} (\mathbf{p}_{t+1})
\]

with \( W_{t+1} := E_{t+1} \ast G_{\text{smooth}} \) where \( \ast \) is the convolution operator. As shown in eq. \( 8 \) \( \beta \) affects the mutual “gravity” between the particles. If \( \beta = 1 \), then the particles are equally distributed and the weights differ only according to \( W_{t+1} \). If \( \beta = 0 \), then the particles’ weights only depend on \( p(z|\theta) \) and \( W_{t+1} (\mathbf{p}_{t+1}) \).

It can be seen that the proposed measurement model is not proper to model independently moving objects because they are supposed to yield a high projection error according to eq. \( 5 \) and a low score according to eq. \( 8 \). So this particle filter “swallows” moving objects and therefore reconstructs only the static part of the scene.

### 2.6 Resampling

In this paper the systematic resampling strategy as described by Hol et al.\(^{12} \) is used. According to the weights of the current particles new particles are generated and diffused in the 3D space. The new particles are normally distributed with variance \( \sigma^2_{\text{diffuse}} \) in an area around the old particles. The resampling step distributes particles preferably in regions with a high likelihood that a scene structure is present. This leads to a clustering of the particles as shown in fig. 1.

### 3. PANORAMIC VIEW

#### 3.1 Cylindric Coordinate System

The 3D space is modeled as a cylindric coordinate system (CCoS). More precisely, this is a cylinder minus two cones. The image plane is replaced by an image cylinder which radius is the virtual focal length \( f_v \) (fig. 2). Given a maximum depth radius \( r_z \), a maximum scene height \( Y_{\text{max}} \), the resolution in \( Z \)- and \( Y \)-direction \( \Delta_z \) and \( \Delta_y \), the virtual image height

\[
H_v = \frac{2 \cdot Y_{\text{max}}}{\Delta_y}
\]

and focal length

\[
f_v = \frac{r_z \cdot H_v}{2Y_{\text{max}}}
\]

can be determined. The angular resolution of the cylinder as well as the resolution in \( Y \)-direction is given by

\[
\alpha = \left(\frac{\Delta_y}{Y_{\text{max}}}\right) \cdot \arctan\left(\frac{Y_{\text{max}}}{\Delta_y \cdot f_v}\right).
\]

Finally, a cartesian 3D point \( (X, Y, Z)^T \) can be expressed in polar coordinates \((\varphi, \phi, l)^T\) with

\[
\varphi = \arctan\left(\frac{X}{Z}\right)
\]
\[
\phi = \arctan\left(\frac{Y}{T}\right)
\]
\[
l = \sqrt{X^2 + Z^2}.
\]

One unit of this discrete space is called a quantum \( Q \). All quantums that are intersected by the line that contains the virtual camera center and the 3D point \((\varphi', \phi', r_z)^T\) define the corridor \( C_{\varphi' \phi'} \). Each pixel in the image cylinder corresponds to a corridor and hence the whole space can be modeled by a set of disjunct corridors (see fig. 3). These quantums are filled with the 3D particles and their unscaled weights until a predefined threshold is reached. This may be a number of time steps or a maximum number of accumulated particles.
3.2 Filling the Space

Because it is desirable to calculate the panoramic view as dense as possible (within reasonable computational time) some more points are filled into the CCoS than just the particles. For that purpose we need the projection \( p = (x, y)^T \) of a particle \( \theta \). In a \( m_{\text{fill}} \times m_{\text{fill}} \) region \( M \) around \( p \) in the image plane the 3D point for every pixel within the mask is calculated. This is done by assuming the points \( \theta' \) that correspond to \( p' \in M \) to have the same depth coordinate \( Z(\theta) \).

Hence the full 3D coordinate is calculated as

\[
\begin{pmatrix}
X' \\
Y' \\
Z'
\end{pmatrix} = Z(\theta) \cdot \begin{pmatrix}
x' - c_x \\
y' - c_y \\
1
\end{pmatrix}
\]

with the known focal length \( f \) and principle point \((c_x, c_y)^T\) of the camera. Afterwards the point \((X', Y', Z')^T\) is transformed into the reference coordinate system. This can be done by accumulating the motion parameters (compare e. g. Pagel\(^1\)). To account for the weight \( \omega(\theta) \), every new particle \( \theta' \) is weighted with the product of \( \omega(\theta) \) and a \( m_{\text{fill}} \times m_{\text{fill}} \) gaussian kernel \( G_{\text{fill}} \) centered at \( p \) with variance \( \sigma_{\text{fill}}^2 \), according to \( p' \in M \):

\[
\omega(\theta') := \omega(\theta) \cdot G_{\text{fill}}(p')
\]

3.3 Visualization of Depth and Texture

The particles are considered as a weighted 3D point cloud. When projecting a particle on an image cylinder it may occur that this particle is occluded by another, closer particle. Usually, the closer point hides the farther point and only the closer point is visible in the image. This will only be a problem, if the closer particle is weakly weighted and likely to be an outlier. In that case the far, strong particle would be hidden by a close, weak particle. However, it would be more reasonable to visualize the points according to their weights, so that the strong point gets the most attention when calculating the texture and depth values for each pixel. Maybe one may think of this problem like looking at a distant swarm of birds through a swarm of flies. The problem with the resampled point clouds from a particle filter is that they are not exactly measured. They are rather scattered around a likely region in the 3D space according to the assumed underlying distribution. The
particles are distributed in a point cloud around, but not at a region in the scene where probably structure is present. This means that there are not necessarily strong image features for which temporal correspondences and hence the 3D structure could be determined reliably. Furthermore, the data can become noisy due to estimation errors of the egomotion and optical flow field as well as noisy image data. These noise sources can hardly be eliminated. Finally, the goal is to determine a depth value and a color for each corridor (and hence each pixel). In the rest of this section a strategy is presented to handle noise and object occlusions by weighting each quantum with a transparency factor based on the accumulated particles.

After the particles were filled into the $K$ quantums, a score for each quantum $Q_k$ can be calculated. First, a particle score

$$s_{p_k} = \max_{i=1, \ldots, N_k} \{\omega_{k_i}\} \quad (17)$$

for quantum $Q_k$ is calculated, which is simply the maximum weight of the unnormalized weights of all particles $\theta_{k_i}$ in $Q_k$. The particle weights must be unnormalized because the number of particles in the particle filter is not constant and hence this would result in bigger normalized weights when only little particles are present and vice versa. Second, a color score

$$s_{c_k} = \frac{1}{\sqrt{2\pi}\sigma_{c_k}} \quad (18)$$

is calculated. This is a gaussian distribution $\text{gauss}(x, \mu_{c_k}, \sigma_{c_k})$ at $x = \mu_{c_k}$ with mean

$$\mu_{c_k} = \frac{1}{N_k} \sum_{i=1}^{N_k} g_{k_i} \quad (19)$$

and variance

$$\sigma_{c_k}^2 = \frac{1}{N_k - 1} \sum_{i=1}^{N_k} \left( g_{k_i} - \mu_{c_k} \right)^2 \quad (20)$$

where $g_{k_i}$ is the image value of the projected particle $\theta_{k_i}$. This causes a stronger weighting of quantums which particles have equal color values. The final, normalized quantum score is

$$s_k = \frac{s_{p_k} s_{c_k}}{\sum_{q=1}^{K} s_{p_q} s_{c_q}} \quad (21)$$

A naive approach for estimating the depth would be to use simply the sum of weighted quantums. But this will not lead to satisfactory results: Imagine a corridor with only two quantums that contain particles, the first and the last quantum in the corridor. Both quantums may have the same weight $\omega_1 = \omega_K = 0.5$, whereas every other quantum is zero-weighted. The mean depth would then be $0.5 \cdot (1 \cdot \Delta_z + K \cdot \Delta_z)$ which is the depth value exactly in the middle of the corridor. Obviously this would be a bad choice, because there is no measurement at all. We would rather prefer a depth value at either $\Delta_z$ or $K \Delta_z$. But still both quantums are very uncertain. In this case we would believe that $\Delta_z$ is the more dominant depth value for the current pixel because it is closer to the virtual camera center. But because of the weakness of the first quantum, the $K$th quantum should be considered anyway. Therefore transparency weights are introduced. The unscaled transparency weight $\tau'$ is calculated as

$$\tau'_k = s_k \cdot \prod_{i=2}^{k} (1 - s_{i-1}) \quad (22)$$

$$= s_k \cdot (1 - s_1) \cdot \ldots \cdot (1 - s_{k-1}) \quad (23)$$

for $k > 1$ and $\tau'_1 = 1$ where $Q_1$ is the innermost quantum. To get the final transparency score $\tau_k$, $\tau'_k$ is scaled:

$$\tau_k = \frac{\tau'_k}{\sum_{q=1}^{K} \tau'_q} \quad (24)$$
\( \tau_k \) considers the score \( s_k \) of \( Q_k \) as well as the whole previous corridor structure from \( Q_1 \) to \( Q_{k-1} \). The factor \((1 - s_{k-1})\) causes a stronger weighting of \( Q_k \) if the previous quantum was weak. Once a strong quantum occurred along the corridor all subsequent quantums will get a weaker weight. The texture value of a pixel is calculated as

\[
u_{\text{texture}} = \sum_{k=1}^{K} \tau_k \cdot \mu_{c_k}. \tag{25}\]

Analogously, the depth value is given by

\[
u_{\text{depth}} = \sum_{k=1}^{K} \tau_k \cdot k \cdot \Delta z. \tag{26}\]

Once \( \nu_{\text{texture}} \) and \( \nu_{\text{depth}} \) are known for each pixel, a panoramic texture image as well as a panoramic depth image can be visualized.

### 4. EXPERIMENTS

To test the algorithm a test ride was done. Therefore a camera with a 60° field of view was mounted in front of a vehicle. For egomotion estimation the approach of Pagel\(^1\) was used. The parameters were set as follows:

- \( \Delta_z = 50 \), \( \Delta_y = 10 \), \( r_z = 3000 \), \( Y_{\text{max}} = 1500 \) (the values depend on the scale of the motion estimation), \( \sigma_f = 2.5 \), \( \sigma_c = 3.0 \), \( \sigma_{\text{fill}} = 2.0 \), \( \sigma_{\text{smooth}} = 2.0 \), \( m_{\text{fill}} = 5 \), \( m_{\text{smooth}} = 5 \) and \( \beta = 0.7 \).

For this test runs a fixed variance \( \sigma_{\text{diffuse}}^2 \) for the resampling was used with \( \sigma_{\text{diffuse}} = 2.0 \). A restriction of a maximum of 2000 particles per time step was implemented to keep the computational effort reasonable. We chose a threshold of 40,000 particles per CCoS. Such a threshold seems reasonable for this data, because the accumulated error of the estimated motion parameters leads to a blurring of the 3D structure when the particles are transformed into the reference CCoS. The coordinate systems are painted in fig. 4. After each panoramic view computation a new CCoS was created at the current position and the particle filter procedure continued. The resulting texture and depth panoramic views are shown in fig. 6 to fig. 11. Images from the scene including the particles are shown in fig. 5. Fig. 6 to fig. 9 show good results for the texture panorama as well as for the panorma depth map. In fig. 6 most of the objects in the scene are recognizable. In fig. 11 one can see how the lamp behind the tree gleams through the tree. This is a result of the transparency calculation. Although the field of views in the panoramic images is quite smaller that 360°, one should keep in mind that the algorithm...
Figure 5. Stills from the test sequence with printed particles. Fig. 4 gives an overview of the scene.
Figure 6. Texture panorama of $CCoS_1$ for $\varphi \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$.

Figure 7. Depth panorama of $CCoS_1$ for $\varphi \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$. The color is encoded from bright (far) to dark (close).

is basically designed for multi-camera applications. So the panoramic view could be extended to $360^\circ$ by simply using several extrinsically calibrated cameras (see e.g. the work of Esquivel et al.\textsuperscript{13}). However, even with a forward looking, monocular camera set we could reach a significant expansion of the field of view.

5. CONCLUSION

A particle filter based method was proposed to estimate the 3D structure of the scene. For the presented results only a monocular image sequence and the according egomotion parameters of the camera were used. The advantage of the particle filter is that there is no need to calculate the 3D position for every single point by triangulation. Instead, triangulation has only to be done for a sparse set of image features whereas the largest set of 3D points is propagated by the particle filter. Additionally, the particle filter implicitly clusters the particles at 3D positions where it is likely to be a scene structure. As a result, a lot of interventions have to be done to guarantee as much 3D information in the field of view as possible. Therefore importance samples were added to balance the particles. The scoring was also manipulated by using the equality factor...
Figure 8. Texture panorama of $CCoS_2$ for $\varphi \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$.

Figure 9. Depth panorama of $CCoS_2$ for $\varphi \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right]$. The color is encoded from bright (far) to dark (close).
Because each particle is weighted the proposed approach is proper to work in noisy environments. This is also advantageous for fusion tasks. The CCoS data structure is basically suitable for parallel processing on multi-core machines. The calculation of the depth and texture value for each corridor respective for each pixel in the virtual image cylinder can theoretically be executed at the same time.

Good results could be reached with real data. Although the accumulation in a reference CCoS is strongly dependent on the quality of the egomotion estimation, this seems to be a solvable problem. There are several strategies in literature that may handle this problem.

The current results can be seen as a fundament for further applications. There are plans for zoom-applications, 3D-mosaicing or virtual guides through the 3D scene. Some problems that arise with these applications are referencing and fusion of several noise-affected CCoS. Therefore self-analysis methods should be implemented. The handling and correction of motion errors over time will be a major topic for further developents.
For multi-camera applications a test rig must be extrinsically calibrated. Currently, developments in online 3D reconstruction, modelling and visualization for mobile rigs with non-overlapping field of views are in progress.

REFERENCES