Windowed Factorization and Merging

B. van den Berg and I. Wanders

Abstract—In this work, an online 3D reconstruction algorithm is proposed which attempts to solve the structure from motion problem for occluded and degenerate data. To deal with occlusion the temporal consistency of data within a limited window is used to compute local reconstructions. These local reconstructions are transformed and merged to obtain an estimation of the 3D object shape. The algorithm is shown to accurately reconstruct a rotating and translating artificial sphere and a rotating toy dinosaur from a video. The proposed algorithm (WIFAME) provides a versatile framework to deal with missing data in the structure from motion problem.

I. INTRODUCTION

In the last two decades substantial progress has been made in solving the structure from motion (SfM) problem. There are several linear methods that describe SfM, including epipolar geometry, the closely related trifocal tensor [1], [2], and factorization. The last method, first outlined in the seminal article by Tomasi and Kanade [3], has been most popular in the last decade since it determines an optimal fit based on all available complete data sequences. Originally, this method was based on an orthographic camera model. This has been extended by Poelman and Kanade [4], by proposing a paraperspective factorization method based on Tomasi-Kanade factorization.

Factorization is frequently used for sparse object-shape reconstruction, including facial 3D reconstruction. Lee et al. [5] use factorization to compute sparse models of a face and used a morphable model to obtain a dense reconstruction. Jo et al. [6] use factorization to optimally estimate sparse facial shape data using several views of a face obtained from a single image using a sparse 3D morphable model (s3DMM).

A drawback of both original factorization methods [3], [4] is that they are sensitive to noise and occlusions. In the work of Tomasi and Kanade [3] these drawbacks are solved by iteratively minimizing the error and filling in the missing data by known values of that point. Noise in the measurements is caused by errors in the tracking of features. A feature that is incorrectly tracked will not only cause an outlier in the reconstructed set of 3D points, it will also bias the estimation of the 3D position of other points. Occlusion of the object makes it impossible to accurately track the occluded points and will result in missing data. Since singular value decomposition cannot deal with missing data, incomplete data sequences have to be excluded in order

to perform the original factorization algorithm as described by Tomasi and Kanade. In addition, the optimization problem solved by factorization tends to have an ambiguous solution (e.g. as with the Necker cube reversal) [7].

To improve performance of factorization on sequences with noise and missing data, more elaborate SfM methods have been developed recently, of which most methods use factorization as a basis. Marques et al. [8] describe a method for direct factorization with degenerate and missing data. Additionally, non-linear batch and recursive approaches to the SfM problem have emerged to deal with these issues. Generally, these techniques directly try to solve the object rotation matrix and projection by error minimization of tracked feature coordinates. Batch techniques include error minimization using non-linear least squares [9] and recursive techniques include sequential depth estimation in each frame and convergence to a model using a Kalman filter [10]. These algorithms offer successful means to deal with noise and missing data but do not yet offer suitable methods for online implementation, since they are designed to process all data in one step.

Relatively little work has been done on online SfM. Mouragnon et al. [11] developed an online algorithm for camera pose estimation using local bundle adjustment. Klein et al. [12] developed an online algorithm for mapping of an environment for augmented reality using a Simultaneous Localization And Mapping (SLAM) formulation of the problem. However, a major difference with this work is that these papers focus on mapping of an environment by a moving camera rather than the mapping of an object with a static camera. In this situation a lot of knowledge can be gained about the motion of the camera with techniques such as visual odometry [13]. Online implementations of the factorization algorithm have been developed by Balzano et al. [14] and with increased robustness to outliers by He et al. [15]. These algorithms use an incremental version of singular value decomposition. Kennedy et al. [16] showed the usefulness of these algorithms for solving the SfM problem online for reconstruction of objects.

In this work, a new method of online SfM that deals with missing and degenerate data with outliers is proposed and evaluated: windowed factorization and merging (WIFAME). The algorithm's 3D reconstruction part, is a direct implementation of the original factorization algorithm by Tomasi and Kanade. However, temporal consistency is exploited in this algorithm to deal with missing data by constraining the factorization to a temporal window. Subsequently, the data of all factorizations is merged in order to compute an accurate estimation of the object's shape.

B. van den Berg and I. Wanders are with the Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, 7500 AE Enschede, The Netherlands. Contact: b.vandenberg-2@student.utwente.nl and i.wanders@utwente.nl.

Both authors contributed equally to this work.

II. THEORY

In this section, the WIFAME algorithm will be outlined. The processing steps are outlined in Figure 1. Each individual processing step is detailed in the next paragraphs.

A. Pre-Processing

For precise object reconstruction, features of the object have to be tracked consistently and with high accuracy. These features should belong to one rigid object. Alternatively, in case of multiple object reconstruction, the separate motion models of the objects have to be identified, as is proposed by Ozden et al. [7]. In this work, the focus is on online single object reconstruction. In order to only track features on the object, a segmentation algorithm is used to identify an object-fitted mask for the feature tracker. Subsequently, points are tracked using a Lucas-Kanade feature tracker [17]. Every tracked feature is labelled with a unique ID. For every frame i , this yields the x and y coordinates in the image for each tracked point l (denoted $x_{i,l}$ and $y_{i,l}$). Additionally, the feature-set is updated in every frame by attempting to identify new features and pruning inconsistently tracked features. To prevent a bias of the reconstruction due to drift of the tracked points, the tracking period of every point is limited to several factorization windows.

B. Windowed Factorization

To deal with sparse data, temporal consistency is assumed and only temporally local data is used as input for the factorization algorithm. Since the factorization algorithm requires a dense data matrix, only those points that were consistently tracked for the entire window w are used for the factorization. However, since only a small temporal window is used, a significant part of the data is conserved. This data is used to generate the matrix $\mathbf{W}$. This data matrix is used as input for the factorization algorithm [3], of which the implementation is described by algorithm 1.

The factorization algorithm computes the 3D positions from $\mathbf{W}$ by projecting its singular value decomposition into the manifold of motion matrices. Subsequently, the relative motion between camera and object is determined, which gives enough information to project the 2D positions from $\mathbf{W}$ into 3D space. Since this is an ambiguous problem, two solutions are possible which are mirrored versions of each other. A comprehensive method to solve this ambiguity is given by Ozden et al. [7]. However, since this work only deals with single object reconstruction, this ambiguity can be solved by regarding the outcome of the first factorization result as ground truth. Subsequently, mirrored factorization results can be corrected when a flip of one or more of the axes is detected, which can be done based on the difference between the axes in the current step and the axes in the previous step.

C. Registration

Every factorization returns a set of 3D points with corresponding ID's as output. Additionally, the quality of the factorization can be estimated based on the ratio between

Algorithm 1: Windowed Factorization [3]. at frame i , with K features, and a window w .

for at every frame i **do**

Generate dense data matrix for (w) frames:

$$\mathbf{W} = \begin{bmatrix} x_{i-w+1,1} & x_{i-w+1,2} & \dots & x_{i-w+1,K} \\ x_{i-w+2,1} & x_{i-w+2,2} & \dots & x_{i-w+2,K} \\ \vdots & \vdots & \ddots & \vdots \\ x_{i,1} & x_{i,2} & \dots & x_{i,K} \\ y_{i-w+1,1} & y_{i-w+1,2} & \dots & y_{i-w+1,K} \\ y_{i-w+2,1} & y_{i-w+2,2} & \dots & y_{i-w+2,K} \\ \vdots & \vdots & \ddots & \vdots \\ y_{i,1} & y_{i,2} & \dots & y_{i,K} \end{bmatrix}$$

Singular value decomposition of $\mathbf{W}$:

$$\tilde{\mathbf{W}} = \mathbf{O}_1 \mathbf{\Sigma} \mathbf{O}_2$$

Estimate quality at this frame, q_i , of singular value decomposition:

$$q_i = \frac{\Sigma_{3,3}}{\Sigma_{4,4}}$$

Restrict to 3D:

$$\mathbf{\Sigma}' = \mathbf{\Sigma}_{1:3,1:3}$$

$$\mathbf{O}_1' = \mathbf{O}_{1:3,\text{all}}$$

$$\mathbf{O}_2' = \mathbf{O}_{2:3,\text{all}}$$

Compute estimates of $\mathbf{R}$ and $\mathbf{S}$:

$$\hat{\mathbf{R}} = \mathbf{O}_1' \sqrt{\mathbf{\Sigma}} \quad \hat{\mathbf{S}} = \sqrt{\mathbf{\Sigma}'} \mathbf{O}_2'$$

Determine real $\mathbf{R}$ and $\mathbf{S}$ using orthometric matrix $\mathbf{Q}$:

$$\mathbf{R} = \hat{\mathbf{R}} \mathbf{Q}$$

$$\mathbf{S}_{\text{local},i} = \mathbf{Q}^{-1} \hat{\mathbf{S}}$$

end

the third and the fourth largest singular values of the singular value decomposition [3]. This value gives an indication how well the first three dimensions of the model explain the variation in the 2D positions of the points. If this ratio is low, a fourth dimension is necessary to explain this variation, and therefore the first three dimensions are not sufficient, indicating non-rigid properties or inaccurate data. In the returned set of points, the coordinates are local coordinates of the factorization. In the registration step, as described in algorithm 2, these local coordinates are converted to the object coordinate system, which is based on the coordinate system of the first factorization. For every consecutive frame we can use the points common to both local factorizations to determine an affine transformation, this handles the rotation, translation and scaling transformations that can occur. In order to prevent a biased estimation due to outliers, RANSAC is used to determine the affine transform between the two point clouds, discarding outliers in the computation of the

Fig. 1. The steps in the algorithm. During a limited time window points are tracked and this yields 2D point coordinates at each frame. Subsequently, the points that were consistently tracked during the time window are used to generate a local reconstruction using Tomasi-Kanade [3] factorization. The local reconstruction is then transformed to the object coordinate system by fitting the reconstruction's previous point cloud to the current point cloud.

transform. With this affine transform all the points from the factorization are converted into the object coordinate system, for each labelled point this results in a position estimate for this point in each frame in which it was tracked. The calculated position of each point in each frame is used in the merge step described in the next section.

D. Merging

During merging as described in algorithm 3, the sparse 3D shape of the object is estimated based on the set of point clouds and information about their quality. There are two issues the merging step has to deal with:

- 1) The factorization algorithm is highly sensitive to noise, and therefore to inaccurately tracked points. Inaccurately tracked points can lead to outliers in the 3D reconstruction.

- 2) The errors of the point's calculated position in each frame are not normally distributed.

The proposed algorithm to merge the points is composed of two steps. Firstly, it uses the quality measure to select only those data points associated with the highest quality factorization, since low quality data does not accurately represent the object's 3D shape. Secondly, the algorithm iteratively converges to the highest point density by excluding points with the largest Mahalanobis distance. This eliminates the outliers and the final estimation is made by averaging the remaining points.

III. RESULTS

The algorithm was evaluated using two test videos, each testing different aspects of the algorithm.

- A translating and rotating sphere: This video shows a translating and rotating sphere with a grid projected on the sphere, and was rendered digitally with a resolution

Algorithm 2: Registration of the reconstruction computed by the factorization algorithm: the current 3D point reconstruction in the local coordinate system and frame i is $\mathbf{S}_{\text{local},i}$, $\mathbf{S}_{\text{previous}}$ is in the object coordinate system.

```

for every  $\mathbf{S}_{\text{local},i}$  points labelled  $L_i$  do
  Determine the points both clouds have in common:
    
$$L_c = L_{\text{previous}} \cap L_i$$

  Let  $\mathbf{C}_{\text{local},i}$  and  $\mathbf{C}_{\text{previous}}$  be the selections from  $\mathbf{S}$ 
  for points in  $L_c$ .
  Estimate affine transformation:
    
$$\mathbf{H}_{\text{local}} = \text{affine\_ransac}(\mathbf{C}_{\text{local},i}, \mathbf{C}_{\text{previous}})$$

  Determine transformation to object coordinates:
    
$$\mathbf{H}_{\text{local} \rightarrow \text{object}} = \mathbf{H}_{\text{local}} \cdot \mathbf{H}_{\text{previous}}$$

  Compute factorization positions in object coordinate
  system:
    
$$\mathbf{S}_{\text{object},i} = \mathbf{H}_{\text{local} \rightarrow \text{object}} \cdot \mathbf{S}_{\text{local},i}$$

  Store the following for use in the next step.
    
$$\mathbf{S}_{\text{previous}} = \mathbf{S}_{\text{object},i}$$

    
$$\mathbf{H}_{\text{previous}} = \mathbf{H}_{\text{local}} \cdot \mathbf{H}_{\text{previous}}$$

    
$$L_{\text{previous}} = L_i$$

    
$$L_{\text{total}} = L_i \cup L_{\text{total}}$$


```

end

Algorithm 3: Merging: l is the unique label per point. $\mathbf{T}$ is a list of point positions.

```

for  $l$  in  $L_{\text{total}}$  do
  Let  $I$  be the set of  $i$  for which  $l$  was present.
  Select the 30 highest scoring frames by quality:
    
$$\mathbf{Q} = \text{sort}(\{q_i | i \in I\}, \text{descending})$$

    
$$J = \{i | \text{for } i \text{ accompanying } \mathbf{Q}_{1:30}\}$$

  Let  $\mathbf{T}$  be the positions of point  $l$  in  $\mathbf{S}_{\text{local},i}$  for
   $i \in J$ .
  Iteratively discard outliers:
  for  $n$  iterations do
    
$$\boldsymbol{\mu} = \text{mean}(\mathbf{T})$$

    
$$\boldsymbol{\Sigma} = \text{cov}(\mathbf{T})$$

    Determine the Mahalanobis distance of each
    point in  $\mathbf{T}$ :
    
$$\mathbf{Y} = \text{sort}(\{\text{mahalanobis}(\mathbf{T}, \boldsymbol{\mu}, \boldsymbol{\Sigma}) | i \in I\}, \text{desc})$$

    
$$\mathbf{T} = \mathbf{Y}_{5:\text{end}}$$
 (discard the 4 with furthest distance)
  end
  Estimate final position for point  $l$ :
    
$$\mathbf{x}_l = \text{mean}(\mathbf{T})$$


```

end

Fig. 2. Visualisation of the digital sphere video used for testing, and the reconstruction of the algorithm.

(a) Stil from the toy dinosaur video.

(b) Merge result.

Fig. 3. Stills from the toy dinosaur video and 3D reconstruction results in two poses.

of 1184x1184 pixels and a framerate of 10 fps. This video serves to test the performance of the algorithm with perfect input frames.

- A rotating dinosaur: This video shows a toy dinosaur that is rotated with a non-uniform speed against a white homogeneous background, and was made using a Panasonic Lumix DMC-G3 camera with a 14-45 lens on 45x optical zoom, with a resolution of 1280x736 pixels and a framerate of 30 fps. This video serves to test the performance of the algorithm with realistic input frames.

In this section, the 3D reconstructions that were created from these video's will be shown. In the discussion, the performance will be evaluated more elaborately.

A. Translating and Rotating Sphere

The translating and rotating sphere was reconstructed in 3D with WIFAME using a window size of 20 frames. During merging quality-based selection of points was made. To exclude outliers, the merging algorithm used 3 iterations and excluded the furthest 4 outliers per iteration. The result is shown in Figure 2. It is shown that the points are located very close to the sphere's surface and spaced similarly to the intersections of the grid in the video.

B. Dinosaur

The rotating dinosaur was reconstructed in 3D with WIFAME using a window size of 50 frames. During merging, the 16 points with the best quality were selected in each point cloud. To exclude outliers, the merging algorithm used 4 iterations and excluded furthest 3 outliers per iteration. The result is shown in Figure 3. Since this is a significantly more complex shape and since the video was made in a real-life situation the reconstruction includes artefacts due to the shape, texture and lighting conditions. The influence of these artefacts is further discussed in the following section.

IV. DISCUSSION

The results section demonstrated successful application of the algorithm for reconstruction of a sphere and a toy dinosaur.

The sphere movie was used as a phantom to test the performance of the algorithm in the optimal situation: providing sufficient trackable points and slow, uniform, object motion. The results with the artificial sphere movie show that the algorithm is capable of accurately reconstructing objects if high-quality input data is supplied.

The dinosaur movie served as a more representative setting to evaluate the algorithm's performance with a complex object, non-uniform movement and real-life lighting conditions. The 3D reconstruction obtained from this movie clearly resembles the dinosaur, except for some of the finer features of the dinosaur. For example, the main body was successfully reconstructed including the finer details such as the leg muscles, while the horns are not visible due to the lack of consistently tracked points.

This demonstrates that the performance of the algorithm in real situations is strongly dependent on parameters with respect to segmentation and tracking.

Therefore, accurate tracking of points on the object is essential for precise object reconstruction. The following are major influences on the tracking precision:

- Segmentation of the object: successful segmentation prevents tracking of points outside of the object, which would violate the rigidity assumption. An alternative solution for this issue is given by Ozden et al. [7], by proposing a method for multi-object reconstruction.
- Lighting of the object: Diffuse lighting prevents shadows and specular reflections, which might cause tracked features on the object to move inconsistently with respect to the object's movements.
- Trackability of the feature: Distinctive isolated features have to be present for accurate and robust tracking.

In this work, the Lucas-Kanade tracker was applied [17], which is generally not robust to specular reflections and depends on a high minimum eigenvalue of the features. A tracker which is more suited to these conditions might improve the performance in less than ideal lighting conditions.

It also holds for other parts of the algorithm, that tuning or replacement might improve the results for a specific situation. In fact, the core idea of WIFAME is that 3D reconstruction is applied over a window of time, enabling online 3D reconstruction of degenerate and occluded data. Therefore, the Tomasi-Kanade factorization in the algorithm might be replaced to improve on local 3D reconstruction. For example, the Tomasi-Kanade factorization should be replaced by Poelman-Kanade factorization to deal with projective effects.

Additionally, the merging step can be adapted for improved performance in specific situations. For example, if there are planes present on the object this could provide additional constraints to improve the factorization result. If any length in the reconstruction is known, this can also be used to overcome the current limitation that the scale of an object cannot be determined.

Besides changing existing steps in the algorithm, additional steps could be included to improve the performance. A major improvement of the algorithms accuracy could be made by including loop-closure in case a previously seen point comes back into view. In its current form, the algorithm would accumulate an error in the reconstruction of an object when the objects movement would contain multiple rotations, since the algorithm does not recognize earlier detected landmarks and does not use this information for improvement of the estimation. In case of loop-closure, multiple object rotations will improve earlier estimations of the shape and ultimately converge to an accurate representation of the 3D shape.

V. CONCLUSIONS

In this work, the proposed WIFAME algorithm was shown to accurately reconstruct a dinosaur phantom. The performance of WIFAME in a specific situation is strongly

dependent on its implementation. The implementation's performance can be adapted by modifying a large set of parameters, including the tracker settings, the window size, the merger setting and the algorithms chosen for each step in the processing pipeline. In this sense, the reconstruction of the dinosaur provides a nice example of one application of the algorithm, but does not cover the extend of applications in which the algorithm could be applied. Furthermore, the large number of parameters makes it hard to compare it with other algorithms. In this work it is shown that with the current parameter set, the algorithm performs well in the reconstruction of diffusely illuminated texturized 3D objects with a smooth background. Therefore, WIFAME is suitable for a broad range of applications such as 3D replication and object classification.

REFERENCES

- [1] R. I. Hartley, "Lines and points in three views and the trifocal tensor", *International Journal of Computer Vision*, vol. 22, no. 2, pp. 125–140, 1997.
- [2] A. Shashua and M. Werman, "Trilinearity of three perspective views and its associated tensor", in *Computer Vision, 1995. Proceedings., Fifth International Conference on*, Jun. 1995, pp. 920–925.
- [3] C. Tomasi and T. Kanade, "Shape and motion from image streams under orthography: A factorization method", *Int. J. Comput. Vision*, vol. 9, no. 2, pp. 137–154, Nov. 1992.
- [4] C. J. Poelman and T. Kanade, "A paraperspective factorization method for shape and motion recovery", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 3, pp. 206–218, Mar. 1997.
- [5] S. J. Lee, K. R. Park, and J. Kim, "A sfm-based 3d face reconstruction method robust to self-occlusion by using a shape conversion matrix", *Pattern Recognition*, vol. 44, no. 7, pp. 1470–1486, 2011.
- [6] J. Jo, H. Choi, I.-J. Kim, and J. Kim, "Single-view-based 3d facial reconstruction method robust against pose variations", *Pattern Recognition*, vol. 48, no. 1, pp. 73–85, 2015.
- [7] K. E. Ozden, K. Schindler, and L. V. Gool, "Multibody structure-from-motion in practice", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 6, pp. 1134–1141, Jun. 2010.
- [8] M. Marques and J. Costeira, "Estimating 3d shape from degenerate sequences with missing data", *Computer Vision and Image Understanding*, vol. 113, no. 2, pp. 261–272, 2009, cited By 36.
- [9] R. Szeliski and S. B. Kang, "Recovering 3d shape and motion from image streams using nonlinear least squares", in *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR '93., 1993 IEEE Computer Society Conference on*, Jun. 1993, pp. 752–753.
- [10] S. Soatto, P. Perona, R. Frezza, and G. Picci, "Recursive motion and structure estimation with complete error characterization", in *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR '93., 1993 IEEE Computer Society Conference on*, Jun. 1993, pp. 428–433.
- [11] E. Mouragnon, M. Lhuillier, M. Dhome, F. Dekeyser, and P. Sayd, "Generic and real-time structure from motion using local bundle adjustment", *Image and Vision Computing*, vol. 27, no. 8, pp. 1178–1193, 2009.
- [12] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces", in *Mixed and Augmented Reality, 2007. ISMAR 2007. 6th IEEE and ACM International Symposium on*, Nov. 2007, pp. 225–234.
- [13] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry", in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 1, Jun. 2004.
- [14] L. Balzano, R. D. Nowak, and B. Recht, "Online identification and tracking of subspaces from highly incomplete information", *CoRR*, vol. abs/1006.4046, 2010.
- [15] J. He, L. Balzano, and A. Szlam, "Incremental gradient on the grassmannian for online foreground and background separation in subsampled video", in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, Jun. 2012, pp. 1568–1575.
- [16] R. Kennedy, L. Balzano, S. J. Wright, and C. J. Taylor, "Online algorithms for factorization-based structure from motion", in *IEEE Winter Conference on Applications of Computer Vision*, Mar. 2014, pp. 37–44.
- [17] C. Tomasi and T. Kanade, "Detection and tracking of point features", Carnegie Mellon University, Tech. Rep. CMU-CS-91-132, Apr. 1991.