Removing Camera Placement Constraints in Shape from Silhouette on Large Acquisition Volumes

Brice Michoud, Erwan Guillou, Héctor M. Briceño and Saïda Bouakaz

Abstract

Shape from Silhouette (SFS) is a technique used to estimate the 3D shape of objects from their silhouette images. SFS uses the intersection of the visual cones of the silhouettes seen by many cameras to estimate a 3D volume that is guaranteed to contain the object. Unfortunately, if one arbitrarily adds a camera whose visual cone does not intersect this volume, the classical algorithm breaks down. We propose modifications to SFS extend the capture volume with the addition of cameras. In this paper, we define different coherency concepts to relax the camera placement constraints, without adding ghost objects when there is only one object in the scene. Finally, we present a real-time system that captures a person moving through many cameras to demonstrate the application and robustness of our method.

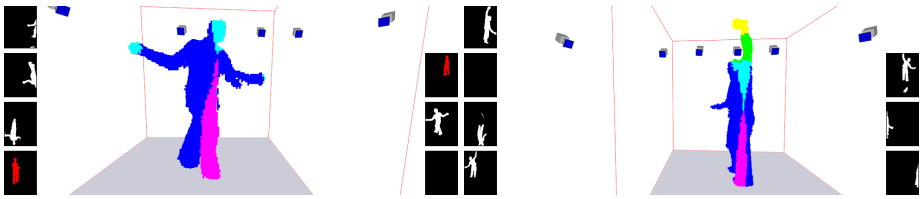

Figure 1: A person traversing different camera's field of view. The colour indicate the number of cameras that see a voxel. There is seven cameras, some which see a full-silhouette (red), some partial-silhouettes, and some nothing at all; The system does not place any constraints in camera positions.

1 Introduction

Capturing dynamic 3D scenes in real-time enables many applications like gesture recognition, behaviour analysis, 3D video, new human-computer interfaces, etc. We wish to perform the real-time motion capture of a person filmed by several calibrated cameras. The user should be unconstrained: the capture system should be markerless and there must be a large acquisition space. Shape from Silhouette (SFS) methods provide a good approach to compute a 3D shape from multiple views, as they are robust and operate in real-time.

SFS methods work by approximating the visual hull of an object. A visual hull is defined as the intersection of silhouette cones from 2D camera views, which capture all

geometric information given by the image silhouettes [1]. One basic property is that a visual hull is the largest volume to be consistent with silhouette information from all views. The reconstructed shape produces same silhouette images as the real object, then it is *silhouette-equivalent* [8]. However, the volumes produced from a SFS reconstruction suffer from a main drawback: the Camera Placement Constraints. The objects that usual SHS method captures must lie in the strict intersection of the field of views of the cameras. Objects that are partially hidden in a certain view, will be cut.

1.1 Related Work

There are mainly two ways to compute an object estimation with SFS algorithms: Surface-based approaches and Volumetric based approaches.

Surface-based approaches compute the intersection of silhouette's cone. First silhouettes are converted into polygons. Each edge is back-projected to form a 3D polygon. Then each 3D polygon is projected onto each other's images, and is intersected with each silhouette in 2D. The resulting polygons are assembled to form an estimation of the polyhedral shape (see [4, 9, 10]). These approaches are not well suited to our application because of the complexity of the underlying geometric calculations. Furthermore incomplete or corrupted surface models could be created, depending upon polyhedron sharpness and silhouette noise.

Volumetric approaches usually estimate shape by processing a set of voxels [2, 6, 12, 11]. The object's acquisition area is split up into a 3D grid of voxels (volume elements). Each voxel remains part of the estimated shape if it's projection in all images lies in all silhouettes. This volumetric approach is well adapted for real-time shape estimation, due to its GPU implementations and robustness to noisy silhouettes.

From the methods that compute a 3D model we note that the classical SFS algorithms require the intersection of all viewing cones. This intersection will form the capture volume. If parts of the subject leave this intersection volume they will not be reconstructed. In our context, this drawback is the most important limitation.

One solution to increase the capture volume is to increase the number of cameras, but the capture volume decreases as the number of cameras increases. These cameras would have to be placed far away to increase the field of view, and would have to have increasing resolution.

Michoud *et al.* [12] proposed an extension of the classical SFS to allow parts of the object to exit the intersection of all cones. Their implementation works in real-time. The main limitation is that connected components of voxels which contains real objects can be removed if none of these voxels is consistent with all the silhouettes. This method fails with the example underlined Fig. 3(b).

Franco and Boyer[5] use a probabilistic 3D representation of scene contents and an occupancy grid. Their method does not suffer or impose any limitation on the camera placement and can reconstruct part of the objects seen by a subset of cameras. There are other methods that do not have restriction on camera placements [7, 13] and can obtain very accurate reconstructions using additional information like colour cues. Unfortunately, these methods rely on heavy statistics, and sometimes need pixel matching and correspondences, which are expensive operations and are far from real-time, thus unsuitable for our applications.

In this paper we propose a Shape from Silhouette algorithm that works in real time and

that does not have any restrictions on camera placement thus permitting a large capture volume.

This paper is organised as follows. In the next section, we present the classical SFS algorithm along with the limitations associated with it. Section 3 presents our approach and how we reduce many ambiguities. In section 4, we provide details about our implementation. Section 5, demonstrates our algorithm under real scenarios. We summarise our contribution and give the perspectives in Section 6.

2 Preliminaries

In this section, we present the standard Shape from Silhouette algorithm (SFS). We introduce the terminology that we will use in the rest of the paper. We then present the limitations and artifacts that exist in classical SFS. In the next section, we propose our solution.

2.1 Classical SFS

Shape from Silhouette algorithms try to deduce the shape of objects from the silhouettes seen by multiple cameras. It is a concept based on the visual hull of an object. We use a volumetric based approach as it is simple to implement and amenable to a GPU implementation. Our terminology is as follows:

- the space is divided into nb_{voxels} voxels ,
- S_i is the silhouette of the real object from the camera C_i with $i \in [1, \cdots, n_{cam}]$,
- v_i is a voxel i in space with $0 < i < nb_{voxels}$,
- n_i is the number of cameras where the voxel v_j can be seen,
- m_i is the number of silhouettes where the voxel v_j projection lies inside.

Thus to classify a voxel v_i under **Classical SFS** which uses the strict intersection of the projected silhouettes, we would test whether the voxel is visible and it is inside the same number of silhouettes and whether this number is *strictly equal* to the number of cameras. According to the above notation, the reconstruction based on SFS using n_{cam} cameras can be written as:

$$SFS = \{v_i, n_i = m_i = n_{cam}\}. \tag{1}$$

This approach has many limitations and artifacts. The most important limitation is related to acquisition space or **capture volume** (see Fig. 2). The objects to capture must lie in the strict intersection of the field of views of the cameras or visual cones seen by the cameras. Objects that leave this space will be cut.

The placement and the video resolution of the cameras will determine the granularity and precision of what can be reconstructed. As the capture volume decreases as the number of cameras increases, it would be difficult to extend thhis volume while conserving a good resolution.

In addition to these limitations, classical SFS algorithms can have two kinds of artifacts:

Using a finite number of cameras we cannot capture the object at every angle. This leads to areas without sufficient information to determine if they belong or not to the

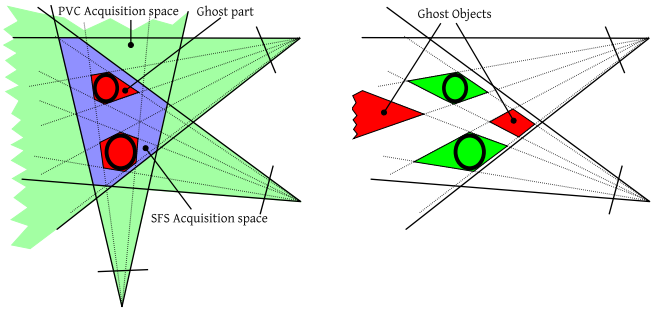

Figure 2: Left picture: the blue region represent the acquisition space with usual SFS. Green part represents the acquisition volume provided by our extensions. Red parts contains phantom parts. They result from missing cameras views (using a finite set of cameras). Right picture : In red, Ghost objects result from ambiguous silhouette information, where the voxels are visually coherent (see §3.1), they do not contain real object.

object. Using the visibility and silhouette tests we can assign these voxels to the object. We call these voxels **phantom parts** (see Fig. 2). Nevertheless, these phantom parts do not pose a problem for many applications like determining the pose of a person.

If there is only one object in the capture volume, the intersection of the silhouettes will lead to one reconstructed object. Alternatively, if there are many real objects in the scene, there will be many silhouettes projected, and these can intersect in empty regions (where there are no objects). The voxels in these regions still satisfy the property: $n_i = m_i = nb_{cameras}$. We call these regions: **ghost objects** (see Fig. 2). We differentiate ghost objects from phantom parts as they are not connected in any way with the real object.

Ghost objects can greatly interfere with many applications of SFS. On the other hand, we accept phantom parts as an inherent artifact of SFS and leave their handling to other methods (see [3]).

We propose to remove the restriction that the visual cones of all cameras must intersect to be able to increase the capture volume. Unfortunately, removing this restriction can lead to the reconstruction of many ghost objects. In the next section, we will present how to remove this restriction, what properties must hold, and how we remove ghost objects. For this paper and for our applications, we consider that there is only one object to be captured.

3 Concepts and Approach

Our goal is to increase the capture space and reconstruct objects using Shape from Silhouette. Furthermore, the minimum requirement of any reconstruction algorithm using Shape from Silhouette, is for the reconstructed model to be *silhouette-equivalent*: the resulting silhouettes of the reconstruction must match those of the cameras. Classical SFS assumes that *all* the camera frustra intersect. We need to relax this constraint in order to enlarge the capture volume without any constraints to camera placements.

In this section, we'll look at two concepts to reconstruct one object using multiple cameras whose visual cones do not have to intersect: Visual Coherence, the minimum

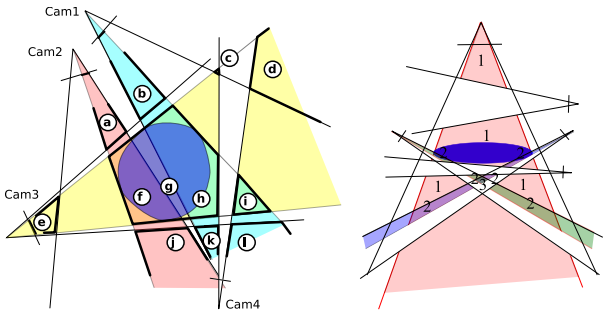

Figure 3: On the left *cam*1 and *cam*2 see partial silhouettes, *cam*3 sees a full-silhouette and *cam*4 sees nothing. $a,\cdots,l$ connected components are visually coherent ($n_i = m_i$) (see §3.1). Connected components c,d,e,i are ghost objects and are not "projective visual" coherent (§3.2). The connex group $f \cup g \cup h$ forms the final resulting object PVC. Note that no region is in more than 2 silhouettes; region c which is a ghost object is also in 2 silhouettes, we cannot infer any notion of quality from the number of silhouettes (§3.1.1). The picture on the right illustrates the notion of quality, *i.e.* number of intersecting cones (numbers shown in figure) is not a sufficient condition to disambiguate the ghost objects from the real object.

requirement to obtain a silhouette equivalent reconstruction; And projective visual coherence, if the reconstruction has many object (all but one are ghost objects), the connected component containing the real object is the only one to be silhouette-equivalent. We now describe each concept in more detail.

3.1 Visual Coherence

The *visual coherency* property helps us satisfy the minimum requirement of any reconstruction, that it be silhouette-equivalent. We satisfy this property by including a voxel in the reconstruction if it belongs to the same number of silhouettes (m_i) as cameras that can see it (n_i). The set VC of voxels which are visually coherent with silhouettes is defined by :

$$VC = \{v_i, n_i = m_i > 0\}. \tag{2}$$

Visual Coherent property is a necessary but not a sufficient condition to know if the real object does in fact occupy this voxel. If the real object occludes the line going from the camera centre, through the voxel toward infinity, the voxel would seem to cover the pixel in the silhouette, thus the voxel could be visually coherent even though the real object does not occupy this voxel. This will result in the reconstruction of many ghost objects in addition to the real object (see Figure 3(a)).

3.1.1 Quality Not Enough

We have selected more voxels than those that belong to the object which results in the creation of ghost objects. Using only the silhouette information we cannot determine directly if they form part of the object or not. Nevertheless we have one hypothesis:

there is only one real object in the scene. With this hypothesis we would like to prune all the ghost objects. We experimented with the concept of voxel quality, the number of silhouettes the voxel belongs to, to filter out ghost objects. The idea was that if a voxel is seen and lies in many silhouettes, we could be more confident that it belongs to the real object. Unfortunately, this measurement is not a good indicator for ghost objects; it is possible for ghost objects to have voxels of higher "'quality"' than the real object (See Fig. 3). The work of Michoud *et al.* [12] is intrinsically based on this quality criteria, and is not a correct solution (*i.e.* ghost objects can be kept while voxels that are containing real object can be removed).

3.2 Projective Visual Coherence

Once we have determined all the visually-coherent voxels, we may still be left with many reconstructed objects – all but one will be ghost objects (assuming only one real object in the scene). In our final step, we group the remaining voxels into connected components.

Let CC_k one of the nb_{CCs} connected component of VC. VC cans be re-written as

$$VC = \bigcup_{1}^{nb_{CCs}} CC_k. \tag{3}$$

The last test to determine if a connected component of voxels corresponds to the real object or to a ghost object is to project the group to all the cameras. The property of "silhouette-equivalent" is used to validate the connected component of VC as containing the real object. We call this *projective visual coherence*.

Proposition

Let CC_k one of the connected component of VC. If there is only one object in the scene, and CC_k is silhouette-equivalent, then CC_k contains the real object.

Proof

Let p a pixel of the silhouette S_j. By construction the only one object projected on the camera C_j contains the pixel p. Let CC_k one of the connected components of VC. If the projection of CC_k on the image C_i does not contain the point p, then CC_k cannot contain the real object.

If the projection into all the cameras C_i of a connected component of VC, is different than the silhouettes, then this connected component is a ghost object. The set of voxels that is Projective Visual Coherent is defined by

$$PVC = \{v_i \in CC_k, \forall l, Proj_{C_l}(CC_k) = S_l\}. \tag{4}$$

where $Proj_{C_l}(CC_k)$ is the projection of the voxels of CC_k onto the image plane of the camera C_l.

In our implementation, we project each connected components of VC using the GPU. The test of silhouette-equivalence is also made on the GPU. This approach is not expensive in time consuming. Figure 3 shows how projective silhouette coherence prunes the remaining ghost objects.

4 Implementation Details

Having removed the camera placement constraints, cameras are calibrated in a specific way. Internal parameters are determined using the classical approach. In contrast, to determine external parameters of each cameras, popular calibration methods [15] assume that the intersection of all camera fields of view are not empty. In our case, we used one camera as a reference, we calibrate a set of cameras using a calibration object, then we move the calibration object within the field of view of our reference camera and calibrated the rest of the cameras. Note that the rest of the cameras can see regions not seen by the reference camera.

The second step consists in silhouette segmentation (see [14] for silhouette segmentation algorithm comparative study). We use the method proposed by [6]. First, we acquire images of the background. The foreground (the human) is then detected in the pixels whose value have changed. We assume that only one person is in the field of view of the cameras, thus this person is represented by only one connex component. Due to camera noise, we can have several connex parts, but the smallest ones are removed: they correspond to noise. We use cameras with 640×480 resolution at 30fps, each camera is connected to a computer that does the silhouette extraction and sends the information to a server.

We estimate the 3D shape of the object filmed using a GPU implementation based on the work [6]. Volumetric SFS is usually based on voxel projection: a voxel remains part of the estimated shape if it projects itself onto each silhouette. In order to find the best way to fit the GPU implementation, we propose to use reciprocal property. We project each silhouette into the 3D voxel grid as proposed in [6]. If a voxel is the intersection of all the silhouette projections, then it represents the original object.

Once we have calculated the visibility and silhouette each voxels belong to, we compute the visual coherency ($n_i = m_i$)on the GPU. Connected groups of VC are computed on the CPU. Finally contribution tests (projective-silhouette coherency) are done using the GPU. As we use a voxel based approach, the re-projection of connected components of voxels onto the cameras cannot be exactly the real-silhouette, due to the sampling. The silhouette equivalence test is then estimated using the number of silhouette's pixels that are overlapped by the projection of each connected components.

Computing shape estimation on large acquisition spaces impose a sampling limitation. Our usual resolution for voxel grid is $128 \times 128 \times 128$ in a box of $2 \times 2 \times 2$ meters, which is an acceptable precision: voxel: 1.56cm per side. With this grid we run at $35 fps$ without colouring the voxels. With a box of 10 or more meters, the precision using this sampling would not be enough. There are multiple solutions: First, increase the voxel grid resolution; this could be very expensive if there is one object in the scene. We could also use a distributed computation to reconstruct the voxels by tiles. Alternatively, we use temporal coherence to compute the shape estimation in a subspace of the acquisition area. At initialisation, we compute a coarse grid over the whole volume. Once we have located the object, we compute a much finer grid around the object. At the current step, the gravity centre of the reconstructed shape is used to place the interest box for the next frame. If the subject is lost (goes outside the acquisition space) the system restarts the initialisation step.

5 Results and Discussion

In the first experiment (see Fig. 4) we compare the standard SFS and our approach. There are 6 cameras of which 4 have a partial view. The classical SFS breaks down because it cannot reconstruct anything outside the intersection of the cameras that capture a full silhouette. In contrast, our algorithm does not have any problems with partial views.

Our next experiment demonstrates each individual test in our algorithm. Figure 5 shows the selected voxels after each step. The subject is fully captured in one camera and is partially captured in five. The "visual coherence" test selects all the voxels that are silhouette-equivalent. The projective-silhouette coherency test eliminates the connected components of voxels that are ghost objects.

The last experiment shown in Fig. 1 and Fig. 6, demonstrates how the system works as the subject straddles different camera's visual cones and is even not seen by some cameras. Fig. 1 the colours encode the number of cameras that see different voxels. The subject can enter and leave different camera cones. The major advantage of this algorithm is that most of computation are made by the GPU, except the connected components selection. Our approach extends successfully the acquisition volume to large capture spaces without sacrificing resolution. With our extensions, the algorithm stills work in real-time (35fps with a voxel grid of $128 \times 128 \times 128$ voxels).

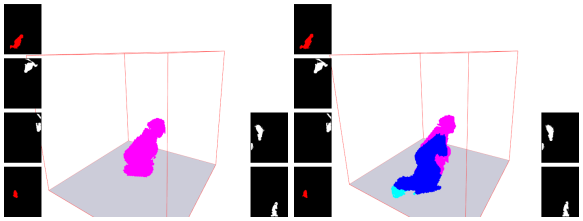

Figure 4: A person kneeling is captured by 6 cameras, 2 cameras see full silhouettes and 4 see partial silhouettes. (left) Parts of the body are missing using Classical SFS . (right) our implementation provide complete results.

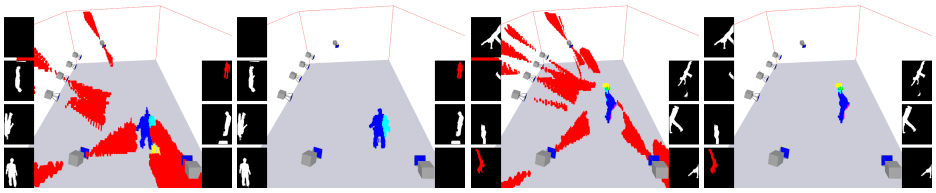

Figure 5: Two different frames (left and right) captured from our framework. The subject may be fully visible (red silhouettes), partially visible (white silhouettes) or not visible by each camera. Simple coherency (top) selects the first set of voxels which are silhouette-equivalent; Full silhouette coherency (middle) removes most ghost objects; Projective-silhouette coherency (bottom) removes the remaining ghost objects which do not receive contributions from the active camera set. The colours encode the number of cameras that see each voxel: red=1, yellow=2, green=3, cyan=4, blue=5, magenta=6, black=7.

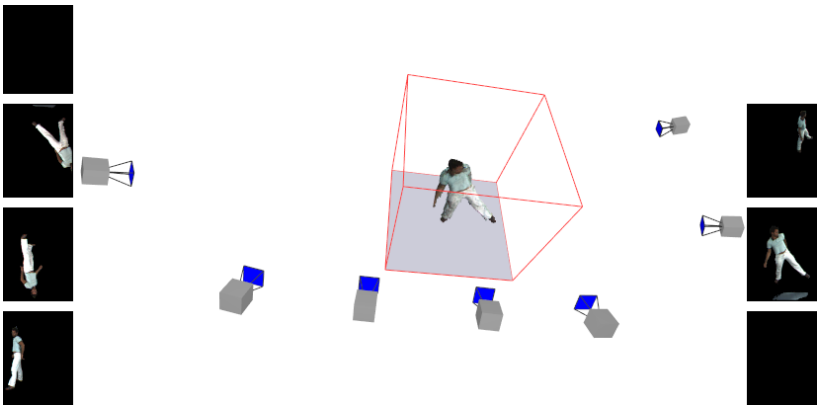

Figure 6: The final result: we can capture dynamic scenes where the subject can move through different cameras. On the borders we see the images captured by the cameras; there is two cameras that does not see the subject.

It is important to note the main limitation of this approach. The visual coherency test is able to work with multiple objects in the scene. But the Projective Visual Coherency test is build from the assumption that there is only one object in the scene. Breaking this assumption will break the Proposition 1.

Our solution is based on the knowledge of the correspondence between the connected components of silhouettes in all images (with only one object, the correspondence is trivial). With multiple objects, we need to compute the correspondences. Having this information, the Projective Visual Coherency will be computed separately for each object (knowing the associated silhouettes), considering that we have only one object to reconstruct. The silhouette equivalence will be tested only on cameras where there is no occlusions.

6 Conclusions and Future Work

In this paper we have presented a contributions that extends the acquisition volume of shape from silhouette approach, without adding ghost objects. Our approach is able to reconstruct 3D shape from object's silhouettes even if cameras see only part or even no part of the object. While most previous approaches assume that the complete silhouette has to be visible, this system is much more flexible in the camera placement, and therefore allows extending the acquisition space. Our system does not place any restrictions on the capture space and on the camera placement. It performs in real time and it is scalable. As we extend the acquisition space, ghost objects appear. With only one object to be captured, we guarantee to remove all the ghost objects. Finally we have proposed a generic solution to remove ghost objects with multiple objects in the scene. The tests we place on each voxel are simple to implement and effective in removing ghost objects.

For future work we are investigating the proposed generic solution, and we are studding what kind of silhouette's connected parts correspondence will better fit with this idea.

References

[1] Bruce Guenther Baumgart. *Geometric modeling for computer vision.* PhD thesis, Stanford, 1974.

[2] G. Cheung, S. Baker, and T. Kanade. Shape-from-silhouette for articulated objects and its use for human body kinematics estimation and motion capture, 2003.

[3] C. Dyer. Volumetric scene reconstruction from multiple views, 2001.

[4] Jean-Sébastien Franco and Edmond Boyer. Exact polyhedral visual hulls. In *Proceedings of the Fourteenth British Machine Vision Conference*, pages 329–338, September 2003. Norwich, UK.

[5] Jean-Sébastien Franco and Edmond Boyer. Fusion of multi-view silhouette cues using a space occupancy grid. In *Proceedings of the 10th International Conference on Computer Vision*, oct 2005.

[6] Jean-Marc Hasenfratz, Marc Lapierre, and François Sillion. A real-time system for full body interaction with virtual worlds. *Eurographics Symposium on Virtual Environments*, pages 147–156, 2004.

[7] Kiriakos N. Kutulakos and Steven M. Seitz. A theory of shape by space carving. *Int. J. Comput. Vision*, 38(3):199–218, 2000.

[8] A. Laurentini. The visual hull concept for silhouette-based image understanding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 16(2):150–162, 1994.

[9] M. Li, M. Magnor, and H. Seidel. Improved hardware-accelerated visual hull rendering, 2003.

[10] C. Liang and K.-Y. K. Wong. Robust recovery of shapes with unknown topology from the dual space. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 2007.

[11] Xin Liu, Hongxun Yao, Guilin Yao, and Wen Gao. A novel volumetric shape from silhouette algorithm based on a centripetal pentahedron model. *icpr*, 1:9, 2006.

[12] Brice Michoud, Erwan Guillou, and Saida Bouakaz. Shape From Silhouette: Towards a Solution for Partial Visibility Problem. In D.Fellner C.Hansen, editor, *Eurographics 2006*, Eurographics 2006 Short Papers Preceedings, pages 13–16, September 2006.

[13] Philippos Mordohai and Gerard Medioni. Dense multiple view stereo with general camera placement using tensor voting. In *3DPVT '04: Proceedings of the 3D Data Processing, Visualization, and Transmission, 2nd International Symposium on (3DPVT'04)*, pages 725–732, Washington, DC, USA, 2004. IEEE Computer Society.

[14] Kentaro Toyama, John Krumm, Barry Brumitt, and Brian Meyers. Wallflower: Principles and practice of background maintenance. In *ICCV (1)*, pages 255–261, 1999.

[15] Zhengyou Zhang. Flexible camera calibration by viewing a plane from unknown orientations. In *ICCV*, pages 666–673, 1999.