Image-Based Multi-view Scene Analysis using *'Conexels'*

Josep R. Casas **Jordi Salvador**

Image Processing Group
UPC – Technical University of Catalonia
Barcelona, Spain

Email: {josep,aljsal}@gps.tsc.upc.edu

Abstract

Multi-camera environments allow constructing volumetric models of the scene to improve the analysis performance of computer vision algorithms (e.g. disambiguating occlusion). When representing volumetric results of image-based multi-camera analysis, a direct approach is to scan the 3D space with regular voxels. Regular voxelization is good at high spatial resolutions for applications such as volume visualization and rendering of synthetic scenes generated by geometric models, or to represent data resulting from direct 3D data capture (e.g. MRI). However, regular voxelization shows a number of drawbacks for visual scene analysis, where direct measurements on 3D voxels are not usually available. In this case, voxel values are computed rather as a result of the analysis on 'projected' image data.

In this paper, we first provide some statistics to show how voxels project to 'unbalanced' sets of image data in common multi-view analysis settings. Then, we propose a 3D geometry for multi-view scene analysis providing a better balance in terms of the number of pixels used to analyse each elementary volumetric unit. The proposed geometry is non-regular in 3D space, but becomes regular once projected onto camera images, adapting the sampling to the images. The aim is to better exploit multi-view image data by balancing its usage across multiple cameras instead of focusing in regular sampling of 3D space, from which we do not have direct measurements. An efficient recursive algorithm using the proposed geometry is outlined. Experimental results reflect better balance and higher accuracy for multi-view analysis than regular voxelization with equivalent restrictions.

Keywords: Multi-view analysis, volume voxelization, epipolar geometry.

1 Introduction

The ever decreasing cost of acquisition devices and computing capabilities are making multi-camera settings increasingly common for visual analysis in controlled environments. Multi-view image analysis exploits similarity and disparity in images provided by multiple cameras observing the scene. This offers promising advantages compared to single camera analysis:

- Multi-view analysis algorithms benefit from 3D cues. Non-redundant information extracted from multiple cameras disambiguates occlusions and augments the available visual information with projections from otherwise occluded parts of the scene

- Multi-view image analysis may also yield additional robustness by redundant detection across views. Object tracking, face detection, gesture analysis, etc. exploit correspondences in the available views by checking the consistency of the analysed primitives (colour, salient points…) in the various projections of the actual 3D scene.

An implicit or explicit auxiliary 3D representation in the form of a volumetric model of the scene is often used as a reference for inter-camera registration in multi-view analysis when camera calibration is available. One usually resorts to an ordered scanning of the 3D space (Cheung 2000, Kutulakos 2000), where volumetric units (or voxels) are equally sized cubes sequentially analysed from their projections in the multiple cameras.

At high resolutions, with the working 3D space sampled at regularly spaced intervals in its orthogonal axes, regular voxelization (Kaufman 1993) is adequate for volume visualization, modelling and rendering of synthetic scenes. Voxelization is also the natural support for data from direct 3D measurements in medical imaging (CT, MRI, ultrasound), biology, geosciences, industry, etc. *However, regular voxelization has a number of drawbacks for multi-view scene analysis*. This is mainly due to the fact that measurements on 3D voxels are not directly available in multi-camera settings. Voxel features are computed rather as a result of the analysis from their projections in multiple views; i.e. the analysis takes place on 'projected' or 'image' data. The actual measurements available are the data sets of pixels belonging to the voxel projections in each view.

The problem arises from the fact that the sampling geometry generated by the regular scanning of the 3D

This work has been supported by the EU through IP CHIL IST-2004-506909 and by the Spanish Government through TEC2004-01914.

space is distorted by the camera projection. Once projected onto the camera images, the sampling geometry becomes irregular, and the amount of data (pixels) from each view available for the analysis of each volumetric unit (voxel) depends on its distance to the camera and on the intrinsic camera parameters. Furthermore, voxel sizes (3D sampling parameters) are not dependent of image resolution and have to be carefully chosen considering the worst case (e.g. projections of two adjacent voxels should not overlap on the same pixel in most of the views). A better approach is to oversample the voxel array so that it can be guaranteed that each 3D sample is drawn from at least a single pixel (Broadhurst 2001).

Figure 1 illustrates this problem. The projections (splats) of one voxel in two different views have varying sizes for cameras located at different distances (this is the usual case for most voxels in the analysed scene). For those views where the splat size is reduced to a few pixels, image data will hardly contribute significant information to the voxel being analysed. Symmetrically, two equally sized voxels project in a different number of pixels on the same camera if they are located at different distances/orientations in 3D space.

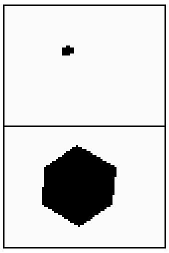

Figure 1 : Two projections of the same voxel, as seen by a close camera (top) and by a far one (bottom)

One way to overcome the dispersion in the image data for voxel analysis is to assign varying weights to the different views when analysing each voxel (Broadhurst 2001). This might result in a certain lack of 'balance' in data sets representing each elementary 3D unit across multiple views. Alternative approaches introduce space discretization which does not rely on regular voxels (Erol 2005) or use hybrid techniques combining volumetric and surface-based approaches (Boyer 2003).

In this paper we follow such alternative approaches and change to an irregular scanning strategy to construct the auxiliary 3D model of the scene under analysis. The resulting geometry is based on the epipolar constraint (Zhang 1998) and is proposed with the aim to better adapt 3D scene analysis to the available image data. It provides a better 'balance' for the analysis of volumetric elementary units from projected data. In addition, the new geometry naturally derives 3D sampling parameters from the original resolution of image data.

The following section analyses regular voxelization and provides statistical values showing the dispersion in the data used to analyse each voxel. Sections 3 and 4 define the proposed scanning geometry and outline a recursive algorithm to scan 3D space. Section 5 analyses statistics of the new geometry and Section 6 compares results obtained for an analysis technique to regular voxelization. Finally, advantages of the proposed method are discussed along with conclusions and future work.

2 Statistics of Voxel Projection Size for Regular Voxelization in Multi-view Analysis

We have analysed the problem of the dispersion in the available image data used to represent each voxel in a particular, albeit common, multi-view analysis situation. In our experiments, a Smart Room is equipped with 5 fully calibrated cameras. Four cameras are placed on the room corners and the fifth one is mounted on the ceiling, providing a zenithal view of the scene.

A regular sampling geometry with 3 cm sided cubic voxels is defined in the 3D working space of 4x5x2 m^3. We have computed the statistics of the projection size (in pixels) for all voxels in the working space. The histogram of the voxel projection size is shown in Figure 2 for one of the corner cameras (cam1). Table 1 outlines minimum, maximum, mean and dispersion values of the voxel projection size for all cameras.

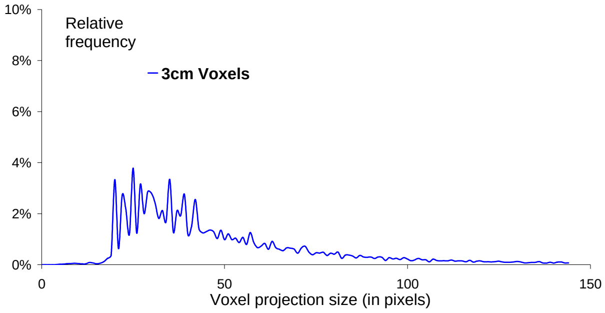

Figure 2 : Histogram of voxel projection size onto camera 1 for all 3x3x3 cm^3 voxels in a 4x5x2 m^3 working space

	Min projection size	*Max projection size*	*Mean projection size*	*Standard deviation in proj. size*
Cam1	4	1797	60	67
Cam2	4	1864	58	65
Cam3	4	1650	59	66
Cam4	4	2155	60	69
Cam5	2	296	40	22

Table 1 : Statistics of voxel projection size (in pixels) for 3 cm sided voxels in a 4x5x2 m^3 working space

The statistics of the voxel projection size show considerable dispersion: standard deviation is of the same order as the mean value. This situation is not favourable for multi-view analysis algorithms when looking for matching visual features in different views or when checking the consistency of the analysis across multiple projections of the actual 3D scene. The analysis algorithm will be using quite different amounts of image data (number of pixels projected in each view) for the analysis of an individual voxel in 3D space. This is due to the dispersion in the projection size of the uniform elementary unit employed in this geometry.

Regular voxelization of 3D space is, therefore, "**non-adapted**" to image data for multi-view analysis. At least in settings as common as a smart room with 5 evenly distributed cameras, voxels project to 'unbalanced' sets of image data in each camera, from which analysis algorithms have to work out feature matches and consistency checks. As mentioned before, one way of adapting the analysis to the available image data would be to avoid giving the same importance to the data set (projection) in each view. When the analysis algorithm has to make a decision on that voxel (e.g. whether it is foreground or background; surface or interior; skin, clothing or object…) it will have to take into account the amount of pixels in each view informing the analysis decision. This strategy depends on the analysis itself and does not solve the lack of 'balance' of the data sets representing each voxel in the different views. The dispersion in the projection data sets is due to an arbitrary 3D scanning geometry chosen to support analysis data.

3 Proposed 3D Scanning Geometry

An alternative strategy is 'image-based' scanning of the 3D scene. Matusik introduced the concept of 'image-based' visual hulls (Matusik 2000, Matusik 2001) to render an observed scene in real time from a virtual camera's point of view without constructing an explicit auxiliary volumetric representation. He claims that the advantage of performing geometric computations based on epipolar geometry in image space is the elimination of resampling and quantization artefacts in volumetric approaches. However, his paper focuses on visualization and rendering applications rather than visual analysis and does not consider the effects of image sampling. We follow Matusik's concept of 'image-based' processing, but focussing on analysis applications. In particular, we propose an image-based recursive scanning algorithm for multi-view analysis, and derive the corresponding geometry in 3D space. This provides a volumetric representation for image data functionally equivalent to regular voxelization as volumetric data support for the analysis algorithms. The 3D scanning procedure is better **adapted** to the image data than regular voxelization, minimizing the dispersion in the amount of data used for the analysis of each voxel in the different views.

The motivation behind the proposed approach is that it does not make much sense to scan the 3D space (from which we do not have direct measurements) with a regular geometry while this results in a non-regular geometry once projected on the camera images. The actual data we have available in multi-view scene analysis applications are visual measurements (pixel data) from the camera images, and we better base the scene analysis geometry on the available data unless there is a clear benefit from not doing so. The proposed procedure changes the usual multi-view analysis paradigm adapting the analysis strategy to the available image data. Instead of scanning 3D space with arbitrary regular voxels –from which we do not have direct data–, the proposed scanning is natural and regular on the camera images, which are divided in a regular way, and the 3D equivalents of such divisions generate the volumetric geometry.

In the next subsections we introduce the basic tools defining a 3D geometry adapted to the images. First, we define the *quadrant* as an image region. Then, the *cone* is obtained as back-projection of the quadrant. The *conexel* –elementary volumetric unit for the proposed geometry– is obtained by intersection of cones. Finally, we outline a recursive algorithm for 3D space scanning in multi-camera settings based on this geometry, which has proven useful for multi-view analysis techniques.

3.1 Image regions: *quadrants*

To avoid dispersion in the amount of image data from the different views used in the analysis of a volumetric unit, we divide camera images in quadrants. *Quadrants* are defined as regular square shaped, non-overlapping regions in the projected images. 3D space scanning will be defined based in the geometry generated by the quadrants, instead of using the voxel-based geometry.

The expected behaviour of the proposed approach is that the data sets in every image will be balanced when scanning a 3D space region: their projections will always lie inside the selected quadrants. Furthermore, the subdivision of the images in quadrants can be made recursive, and the scanning algorithm described at the end of this section exploits this possibility.

3.2 Back-projection of quadrants in 3D: *cones*

The *cone* is the 3D back-projection of a 2D quadrant, also known as the projective extrusion of the 2D silhouette (Matusik 2000). To obtain the 3D back-projection of a quadrant in an image, we compute the back-projected ray of the four corners of the quadrant (Garcia 2005). Then, we compute the inequations of four planes by combining the four ray equations, so that the pixels in the quadrant are the projection of the inner volume enclosed by the four planes. The *Center Of Projection* (COP) of the camera is the main vertex of the cone, which results in a pyramidal shape without a basis.

An illustration of two such cones computed from their corresponding quadrants is shown in Figure 3 for the actual settings of cameras 1 and 2 in our smart room.

3.3 Intersection of two or more cones: *conexels*

The elementary volumetric unit in our scanning geometry is called *conexel*[1]. We define the conexel as the 3D intersection of back projected cones. The cones defining a conexel are generated by a selected quadrant in each available view. The procedure to obtain a conexel is:

1. Select a quadrant for every available camera image
2. Compute back-projected cones for the selected quadrants (a set of 4 inequations define each cone)
3. Obtain volumetric intersection of computed cones

Figure 4 presents a 3D view of a conexel obtained as the intersection of the two cones shown in Figure 3, corresponding to quadrants (2,1) and (1,1) selected from the views in cameras 1 and 2, respectively. Clearly, the geometry of the conexel is that of a polyhedral visual hull and its 3D computation is perhaps not so straightforward. We will see that the defined geometry will be implicitly used in the proposed scanning algorithm and, unless an explicit volumetric representation with conexels is required, multi-view scene analysis algorithms do not need to compute the 3D conexels. Anyway, computing and rendering a 2D 'view-dependent' representation of a polyhedral hull can be done efficiently (Matusik 2001).

[1] Named after 'cone element' in analogy with pixel from 'picture element' and 'voxel' from 'volume element'

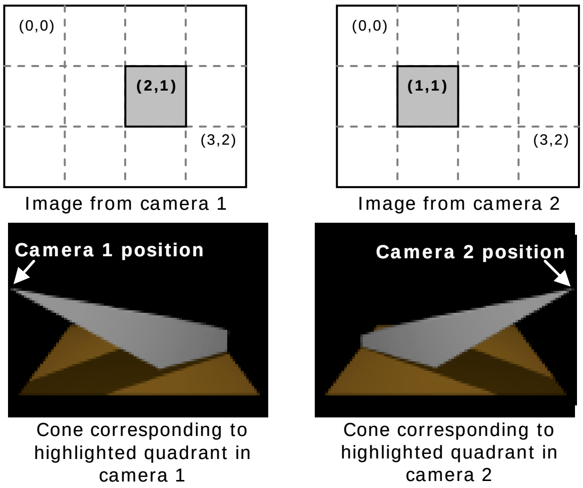
(0,0)
(2,1)
(3,2)
Image from camera 1
(0,0)
(1,1)
(3,2)
Image from camera 2
Camera 1 position
Camera 2 position
Cone corresponding to highlighted quadrant in camera 1
Cone corresponding to highlighted quadrant in camera 2

Figure 3 : Two *cones* (bottom) computed as projective extrusions of the two camera *quadrants* (highlighted at top). Room floor (colored square) included as visual reference

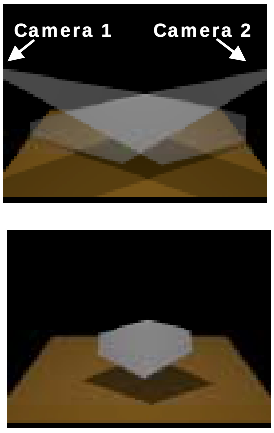

Figure 4 : *Conexel* (bottom) obtained as the intersection of two *cones* (top). Room floor is again included as reference

4 Scanning Method

For multi-view scene analysis purposes, the projection of a conexel in every camera can be computed in a fast way, without having to calculate its actual 3D geometry and project it on every camera image. This is accomplished using epipolar geometry.

In particular, we compute equations of the epipolar lines corresponding to the corners of each quadrant in the other views using fundamental matrices (Hartley 2000). The equations of the epipolar lines are converted to inequations (Ma 2003) so that we can define two image regions in the current view for every cone generated by the quadrant in another view: pixels lying inside the cone projection and pixels lying outside. The area limited by the inequations generated by all epipolar lines defines the projection of the intersection of the cones generated by the quadrants from the other views in the current view (see Figure 5). Pixels inside the quadrant in the current view for the working camera are checked and only those also lying inside the projections of all cones are selected as belonging to the projection of the conexel on the current camera image.

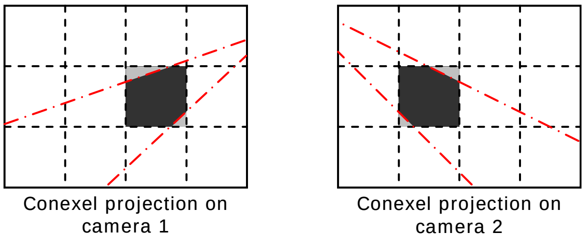
Conexel projection on camera 1
Conexel projection on camera 2

Figure 5 : In general, the reprojection of a *conexel* (dark grey) does not completely cover the generating *quadrants* (light grey). The innermost epipolar lines (corresponding to the corners of the quadrant in the other view) are shown in red for each view

Please note that, when projecting the conexel obtained as the intersection of cones onto the camera images, the projections do not completely cover the quadrants which generated the conexel, as shown in Figure 5. As a consequence, when using the proposed geometry with the conexel as elementary scanning volume, there will still be some dispersion in the amount of image data used for the analysis of the volumetric unit in 3D space. Anyway, the dispersion is expected to be smaller than with regular voxelization. In fact, the number of pixels of the projection of the conexel in each camera view will range from 1 to the total number of pixels in the quadrant, with a dispersion range usually much smaller than the dispersion range for regular voxelization computed in section 2. We will present statistics to assess this statement in a quantitative manner in the results section, proving that the proposed image-based 3D geometry is better adapted to the image data, and provides a better balance in the sets of image data (pixels) characterizing the volumetric elementary unit across the available views.

4.1 Recursive scanning and the *m-tree*

The proposed scanning method based on quadrants can be implemented in a recursive 3D space scanning algorithm, allowing progressive scene analysis approaches. By performing a quad-tree decomposition on the projected 2D image data, each quadrant can be subdivided in four sub-quadrants. The algorithm proceeds by dividing the resulting quadrant in sub-quadrants until some analysis condition is met (e.g. until a foreground or colour consistency check yields true). For each division, the result will be a new set of conexels, always included in the previous one. This strategy can be used to selectively enhance the resolution of 3D analysis only in the regions where needed –such as objects contours– without using the highest resolution in homogeneous space regions, where it is not necessary to subdivide further. Therefore, a progressive space analysis algorithm based on the proposed procedure may start the analysis at rough resolution levels using large quadrants (conexels) and progressively refine the analysis by recursive subdivision to scan at higher resolution only those quadrants (conexels) where needed, depending on the analysis results at the previous resolution level.

Figure 6 illustrates the recursive subdivision and its representation in an m-tree. The *m-tree* (Lu 1996) has been chosen to store the progressive analysis results of the recursive scanning algorithm. Its implementation includes a set of functions which allow moving up, down and sideways in the tree structure.

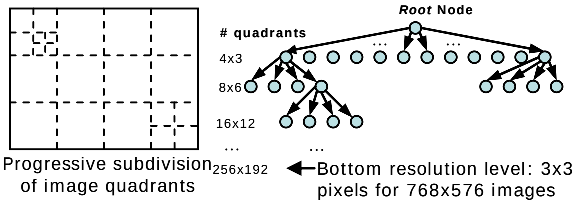
Root Node
# quadrants
4x3
8x6
16x12
...
256x192
Bottom resolution level: 3x3 pixels for 768x576 images
Progressive subdivision of image quadrants

Figure 6 : Progressive scanning of 3D space by recursive subdivision of quadrants and its representation in an m-tree

4.2 Algorithm allowing 3D recursive scanning for progressive multi-view analysis

As the number of cameras may vary, the proposed algorithm will loop depending on the number of cameras. A vector is defined to store the current quadrant under analysis for every camera and a variable stores the current camera in use. The main recursive algorithm to scan 3D space based on the geometry defined by the conexels works through the following steps:

1. Set the current camera view to zero (first camera) and the chosen quadrant vector to all zeros.
2. If the currently selected camera index is larger or equal to zero go to the next step; otherwise, finish.
3. For each camera with smaller index than the one currently selected, select one quadrant and obtain their cone projections onto the currently selected camera view. Also obtain cone projections for the current quadrant in the currently selected camera onto the camera views with smaller index. If any of the cameras does not have any pixel belonging to the conexel projection, it means that no conexel exists for the current set of quadrants. In that case jump to step 6; otherwise, go to the next step.
4. If the currently selected camera is the last one available (meaning that a conexel exists for the current set of quadrants), count the number of pixels of the conexel projection on every camera and, if different from zero, go to next step. In any other case, select next camera and jump to step 2.
5. At this step **any visual analysis function** can be implemented requiring a multi-view consistency check on the projected pixels corresponding to the obtained conexel in 3D. In case that the consistency check needs a higher resolution, jump to step 7; otherwise, store the results in the *m-tree* and go to the next step.
6. If the current quadrant in the current camera is not the last one, increment it. Otherwise, set it to 0, decrement the currently selected camera index and repeat this step while the current camera is larger or equal than zero. Finally jump to step 2.
7. Subdivide each quadrant in smaller quadrants in every view, go down in the *m-tree*, call recursively this procedure and go up in the m-tree again[2]. Then jump to step 6.

5 Statistics of Conexel Projection Sizes for the Proposed Scanning Geometry

As stated before, an improvement of analysis results is expected due to the fact that the presented scanning approach is more natural to image data. In particular, the proposed geometry minimizes the dispersion in the number of pixels used for the analysis of the elementary volumes when projected onto the different camera views.

As the projections of the conexel onto the camera images do not completely cover the quadrants, the projection size of the elementary volumetric unit of the proposed geometry is not constant. We have compared the distribution of the conexel projection size in the same 3D working space with those of regular voxelization shown in section 2. In order to compare the statistics of the two geometries, we note that the average projection size for 3 cm sided voxels is 60 pixels for camera 1 (see Table 1). This value is in between the projection sizes of 6x6 pixels and 12x12 pixels quadrants. This is why we show the distribution for these two cases in Figure 7.

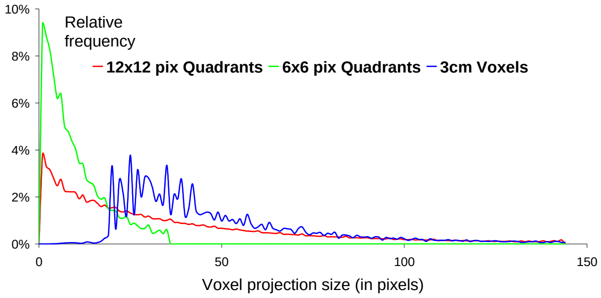

Figure 7 : Histogram of voxel projection size for 3 cm sided voxels (blue line, same as Figure 2), quadrants of 6x6 pixels (green) and quadrants of 12x12 pixels (red)

The distribution of the conexel projection size has completely changed its shape compared with regular voxelization. The range of possible values is restricted to the maximum quadrant size[3] and dispersion values are reduced with respect to the regular case, but standard deviation is still of the order of the mean value of the distribution.

6 Experimental Results

In this section we provide an objective validation proving that the proposed geometry is better adapted to image data in terms of sampling accuracy. This will serve as a proof of concept aiming to quantitatively evaluate the extent to which the geometry based on conexels improves analysis applications in multi-camera settings. Then, we illustrate the progressive capabilities of the proposed multi-view scanning algorithm in a real application.

The analysis application chosen for the experiments is 3D foreground segmentation or Shape-from-Silhouette (Landabaso 2005), which has been designed for 3D object tracking in the smart room.

[2] The available quadrants in every camera are stored in the *m-tree*

[3] Note: For these measures, we assume that the recursive algorithm goes always down to the highest resolution (either 6x6 or 12x12 pixels) for all quadrants in all views.

In Shape from Silhouette applications input data (camera views) are binary images obtained as foreground segmentation masks by a 2D foreground extraction algorithm (Stauffer 2000) from the original camera views. In experiments with real image data, inaccuracies of 2D foreground extraction[4] might prevent an exact quantitative evaluation of the performance of the proposed geometry. This is why we have first chosen the projections of an ideal object (a sphere) in order to have the ground truth available for quantitative comparison.

6.1 Proof of concept: synthetic sphere

For this proof of concept, we have generated 5 simple synthetic scenes. A sphere with a diameter of 1 meter is placed at 5 different positions in the working space of the smart room. As ground truth, we generate the images for the camera views by projecting with the actual intrinsic and extrinsic parameters of every camera. An example of one of these input projections is illustrated in Figure 8.

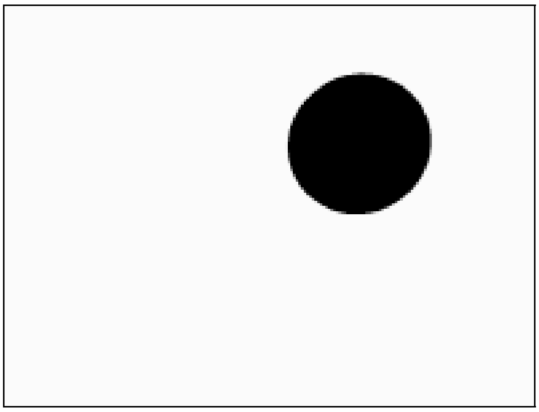

Figure 8 : Input projection for camera 2 for the first of the 5 generated scenes with a simple synthesized sphere. We will take this as ground truth because, contrary to real foreground scenes, it does not show noise effects, misses or false detections

We compare the presented image-based scanning approach for 3D foreground segmentation algorithm with regular voxelization as the competing method. As before, the criteria for comparing results is using a voxel side that, in average, has a projection size in pixels on camera images equal to the number of pixels of the smallest quadrant used in the reconstruction. The formula

$$\begin{aligned} nPix &\approx \pi r^2 \\ &\approx \pi\left(\frac{\sqrt{3}/2\cdot voxSide\cdot K_0}{dist\left(voxCenter, COP\right)}\right)^2 \\ &= \alpha\cdot voxSide^2 \end{aligned}$$

allows computing an equivalent voxel side from the number of pixels we want the voxel to be projected to. We approximate the average voxel projection size in pixels by the projection size of the central voxel in the sphere, and compute the α in the formula for all the scenes and for every camera. Then, an average of α is computed for every scene along all the available cameras.

	Equivalent voxel side for quadrants of		
Synthetic sphere scene num. (average α)	*3x3 pixels*	*6x6 pixels*	*12x12 pixels*
Scene 1 (α=8.256)	1.04 cm	2.088 cm	4.176 cm
Scene 2 (α=8.222)	1.05 cm	2.092 cm	4.185 cm
Scene 3 (α=8.160)	1.05 cm	2.100 cm	4.201 cm
Scene 4 (α=8.226)	1.05 cm	2.100 cm	4.184 cm
Scene 5 (α=7.104)	1.03 cm	2.251 cm	4.502 cm
Voxel side size taken:	**1 cm**	**2 cm**	**4 cm**

Table 2 : Equivalent voxel size side for various quadrant sizes

For quadrants of 3x3 pixels, the equivalent voxel side in all scenes is around 1.1 cm. So, to be in the safe side, we take 1 cm as the equivalent voxel side. This is done similarly for quadrants of 6x6 and 12x12 pixels, as shown in Table 2, resulting in equivalent voxel sides of 2 cm and 3 cm respectively, with a slight advantage in resolution for regular voxelization in all cases. After performing the analysis with both methods, the 3D foreground volume reconstructed is projected back to all cameras.

To evaluate the distortion in the projected image introduced by the sampling 3D geometry with respect to the original ground truth, we define the following metric:

$$dist(rec, gt) = \frac{area(rec \cup gt) - area(rec \cap gt)}{area(gt)}$$

that is, the distance is computed as number pixel differences among reconstructed projection and ground-truth divided by the number of pixels of ground-truth. This distance function is computed for every available projection of the sphere.

In the case of 3x3 pixel quadrants, Table 3 and Table 4 list the distance to ground truth for all 5 scenes and 5 cameras for regular voxelization and image-adapted scanning. Results are given in %, with a multiplicative factor of 100 to render them more easily readable.

	Scene 1	*Scene 2*	*Scene 3*	*Scene 4*	*Scene 5*	*Average*
Cam1	7.40	7.28	7.65	7.57	7.46	7.47
Cam2	7.25	7.46	7.62	7.74	7.34	7.48
Cam3	7.68	7.63	7.46	7.24	7.52	7.51
Cam4	7.62	7.65	7.39	7.49	7.35	7.50
Cam5	7.31	7.46	7.35	7.38	7.47	7.39
Average	7.45	7.50	7.49	7.48	7.43	**7.47**

Table 3 : Distance to ground-truth for regular voxelization with 1 cm sided voxels

	Scene 1	*Scene 2*	*Scene 3*	*Scene 4*	*Scene 5*	*Average*
Cam1	2.03	2.55	3.12	2.91	2.56	2.63
Cam2	2.51	1.95	3.13	2.98	2.83	2.68
Cam3	3.29	2.97	1.89	2.37	2.76	2.66
Cam4	2.74	3.19	2.50	2.01	2.81	2.65
Cam5	2.36	2.37	2.66	2.74	2.52	2.53
Average	2.59	2.61	2.66	2.60	2.70	**2.63**

Table 4 : Distance to ground-truth for image-adapted scanning with 3x3 pixel quadrants

[4] like misses or false detections, and the presence of noise appearing as isolated or grouped foreground pixels

Image-adapted scanning provides more accurate reconstruction. The averaged pixel differences (or errors) are about 35% of those obtained for regular voxelization with quadrants of size 3x3 pixels.

To illustrate those results, Figure 9 shows pixel differences between the ground-truth image and the projection from the reconstructed 3D object for the third scene projected on camera 2 obtained with regular voxelization. Figure 10 is the equivalent result with image-adapted scanning with conexels. Please note how the dispersion in the amount of data used for analysis in regular voxelization causes larger false volumes to appear.

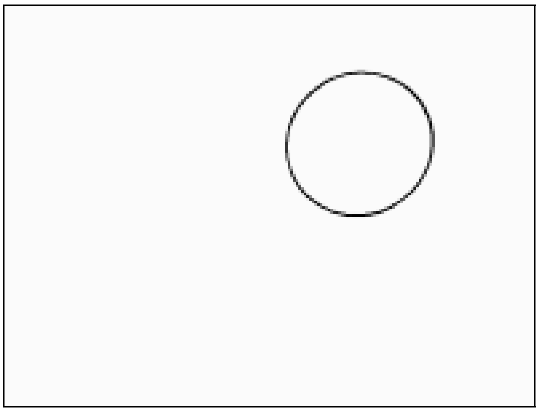

Figure 9 : Pixel differences between the reconstruction with regular voxelization and ground-truth for the third scene on camera 2 (voxel side 1 cm)

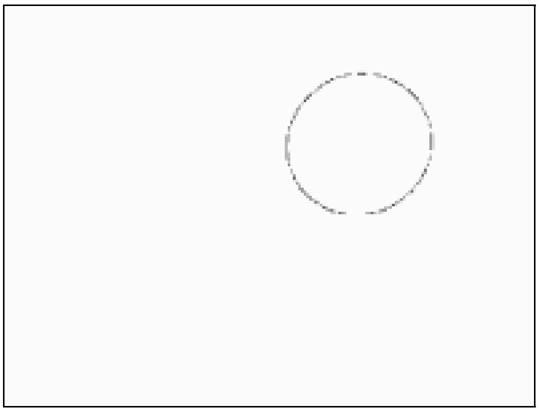

Figure 10 : Pixel differences between the reconstruction with image-adapted scanning and ground-truth for the third scene on camera 2 (3x3 pixels quadrants)

The cases of 6x6 and 12x12 pixel quadrants yield similar results. We have just provided the averaged metrics per scene for two pairs of equivalent cases in Table 5 and Table 6 for regular voxelization and image-adapted scanning. Please observe that, at these lower resolutions, image-adapted scanning still provides more accurate reconstruction. Pixel differences are now about 50% of regular voxelization. Figure 11 through Figure 14 show pixel differences between ground-truth and projection of the reconstructed sphere for regular and image-adapted scanning for the cases of 2 cm and 4 cm sided voxels, and the equivalent cases of 6x6 pixels and 12x12 pixels quadrants.

All cameras	*Scene 1*	*Scene 2*	*Scene 3*	*Scene 4*	*Scene 5*	*Average*
Regular with 2 cm sided voxels	13.23	13.13	13.12	13.11	13.03	**13.12**
Image adapted with 6x6 pixels quadrants	6.31	6.68	6.31	6.47	6.86	**6.53**

Table 5 : Distance to ground-truth for regular voxelization with 2 cm sided voxels and image-adapted scanning with (equivalent) 6x6 pixels quadrants

All cameras	*Scene 1*	*Scene 2*	*Scene 3*	*Scene 4*	*Scene 5*	*Average*
Regular with 4 cm sided voxels	24.21	24.22	24.12	24.18	24.03	**24.15**
Image adapted with 12x12 pixels quadrants	14.34	14.43	13.17	13.98	14.66	**14.11**

Table 6 : Distance to ground-truth for regular voxelization with 4 cm sided voxels and image-adapted scanning with (equivalent) 12x12 pixels quadrants

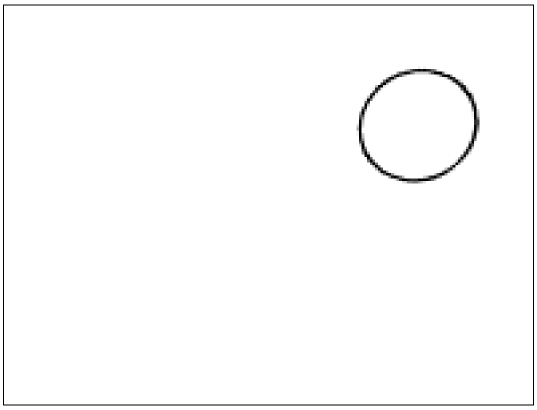

Figure 11 : Pixel differences for the reconstruction with regular voxelization (voxel side 2 cm)

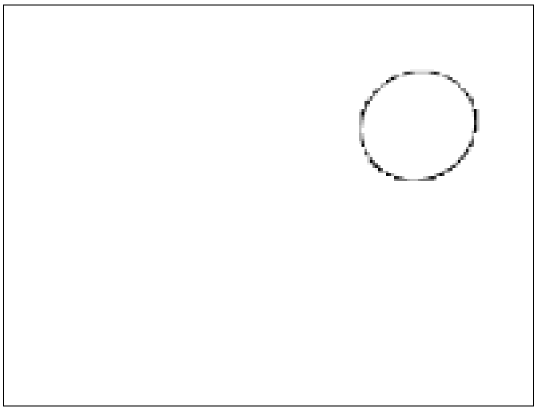

Figure 12 : Pixel differences for the reconstruction with image-adapted scanning (6x6 pixels quadrants)

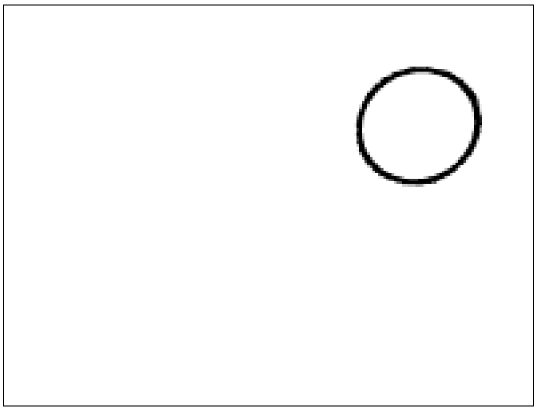

Figure 13 : Pixel differences for the reconstruction with regular voxelization (voxel side 4 cm)

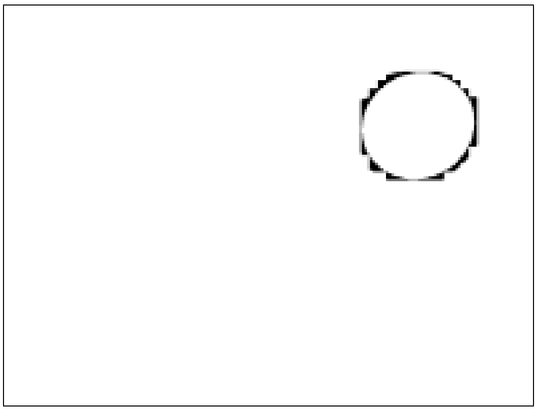

Figure 14 : Pixel differences for the reconstruction with image-adapted scanning (12x12 pixel quadrants)

6.2 Real SfS application

The proof of concept provided above for the reconstruction of a synthetic sphere from its silhouette projections has shown quantitative improvements for the image-based scanning method. We now show qualitative results for the proposed 3D scanning geometry with actual images from our smart room, in a real application of the Shape-from-Silhouette multi-view analysis algorithm. In addition, we illustrate the progressive performance of the multi-view scanning algorithm introduced in section 4.2 in 3D foreground segmentation for object tracking (Landabaso 2005).

Figure 15 and Figure 16 show the re-projected masks obtained as result of 'progressive' Shape-from-silhouette reconstruction by checking foreground consistency in 5 cameras with increasing resolutions in quadrants sizes. Please note that we start at the lowest resolution with only 4x3=12 quadrants of 192x192 pixels each. We apply the consistency check in all views for each conexel, as proposed in the Shape-from-Silhouette technique (Landabaso 2005), to see whether the given conexel is all background, all foreground or mixed. Only in the later case when the conexel is partly foreground and partly background, we continue the recursion at lower resolution by subdividing the quadrant to the next resolution step. As soon as a conexel is detected as uniform (either all foreground or all background), the progressive analysis stops. In these settings, most conexels are only background and this efficiently saves further consistency analysis for such 'uniform' conexels.

Figure 17 shows the results of a different progression of the 3D scanning algorithm also for Shape-from-silhouette reconstruction. In this case, we increase the number of cameras, starting from an initial reconstruction with 2 cameras and then adding the rest one by one. Of course, the two dimensions of progressive analysis (increasing resolution, increasing number of cameras) can be combined at will. The *m-tree* data structure has proven to be a valuable tool to store the data in progressive analysis strategies.

7 Conclusions and Future Work

We have presented an image-based multi-view analysis approach using a 3D space scanning geometry which is adapted to the images. Instead of exploring 3D space (from which we do not have direct measurements, but only projections) with regular geometry, the proposed scanning procedure defines a geometry based on image quadrants. The geometry builds on the concepts of image quadrant, its volumetric extrusion (the cone) and the intersection of two cones (the conexel). This strategy adapts the multi-view analysis to the available data (pixels in camera images), improving the accuracy of the analysis from the multiple views. Contrary to the arbitrary choice of a voxel size in regular voxelization, the sampling geometry in 3D is naturally derived from the resolution of the camera images. Furthermore, volumetric scanning can be progressively refined as the analysis proceeds.

The results obtained show less dispersion in the data sets from the multiple views used to inform analysis decisions for each elementary volumetric unit. As a drawback, we must remark that dispersion in the amount of data used in analysis has not been completely cancelled, but it is more controlled than with regular voxelization techniques. The results also show increased spatial accuracy when compared with regular voxelization. This is the expected behaviour because of the balanced usage of the directly measured data. With the proposed geometry, we do not have to select a voxel size for the working space depending on the smallest splat in the projection of the elementary voxels. The size of the elementary volumetric unit is a consequence of the analysis of image-data (the smallest quadrant).

Furthermore, a recursive algorithm based on the proposed 3D scanning geometry has been described. An interesting feature of the proposed algorithm is its capability for progressive analysis, either by adaptively increasing spatial resolution (subdividing quadrants from larger to smaller sizes when needed), or by adding new cameras to the analysis as their views are made available.

The main directions for future improvements must focus in the study of the connectivity of neighbouring conexels, and in how to use connectivity to remove inner conexels from analysis results in case of need. Another line of study is the set of situations in which conexels are defined from a smaller number of cameras to deal with cases where a conexel is only visible in a subset of all the available cameras.

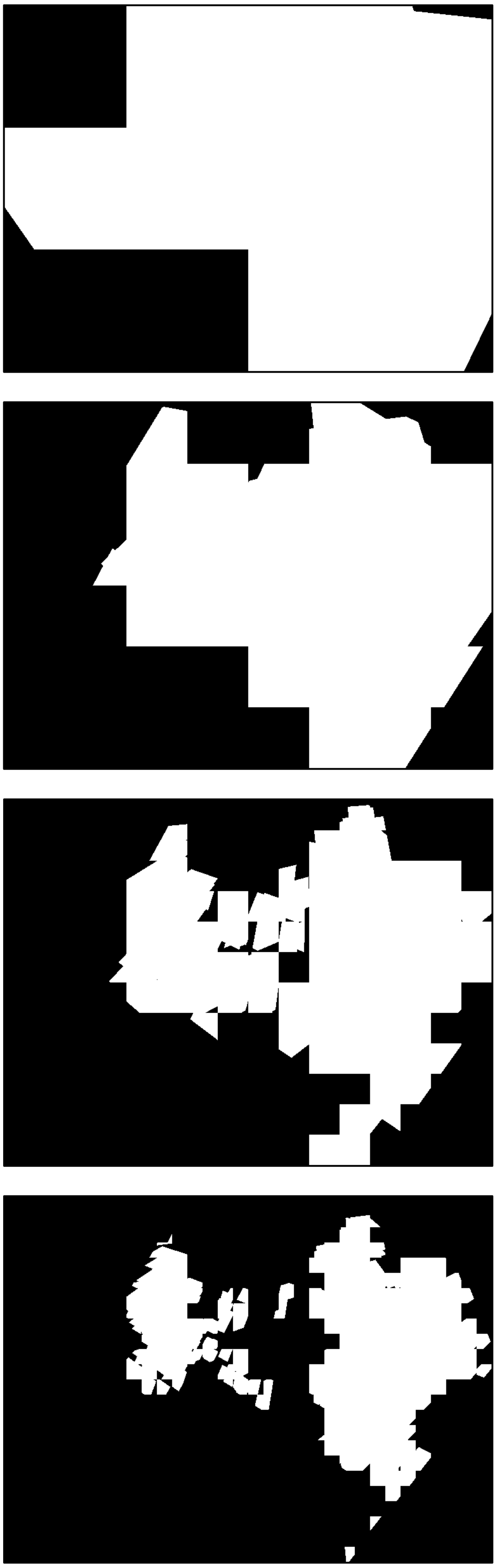

Figure 15 : Results of progressive Shape-from-silhouette reconstruction by checking foreground consistency in 5 cameras with increasing resolutions in quadrants sizes. From top to bottom: 192x192, 96x96, 48x48 and 24x24 pixels per quadrant (continued in Figure 16)

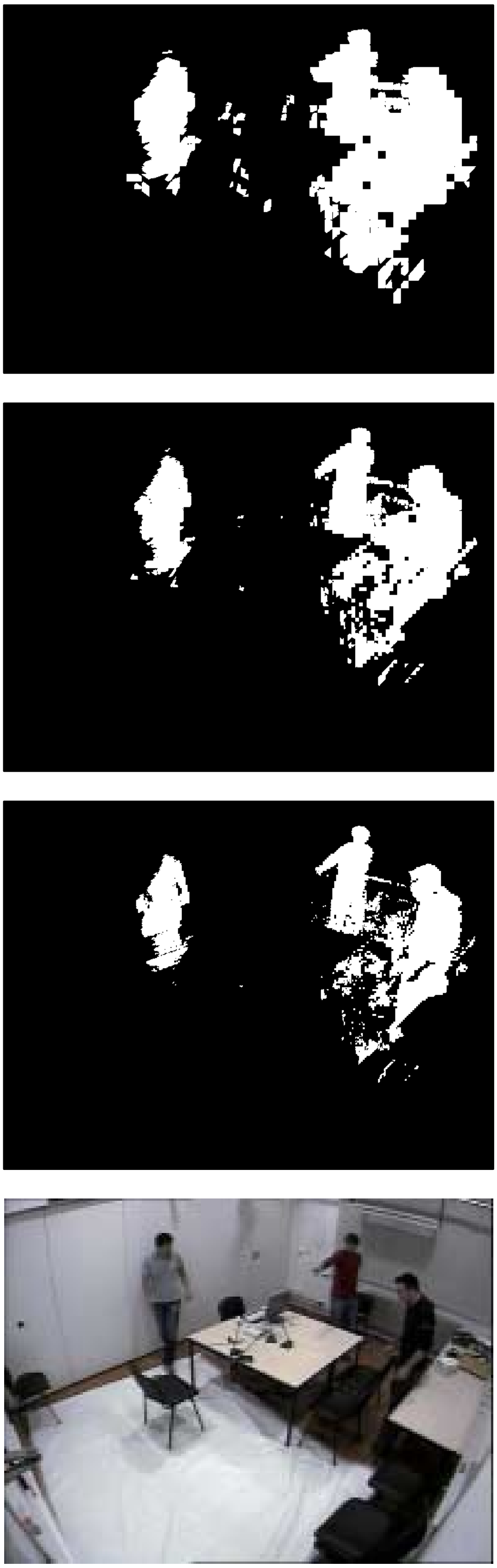

Figure 16 : (continued from Figure 15) Results of progressive Shape-from-silhouette reconstruction by checking foreground consistency in 5 cameras with increasing resolutions in quadrants sizes. From top to bottom: 12x12, 6x6 and 3x3 pixels per quadrant. The last image is the original image with its original lens distortion (corrected in a pre-processing step).

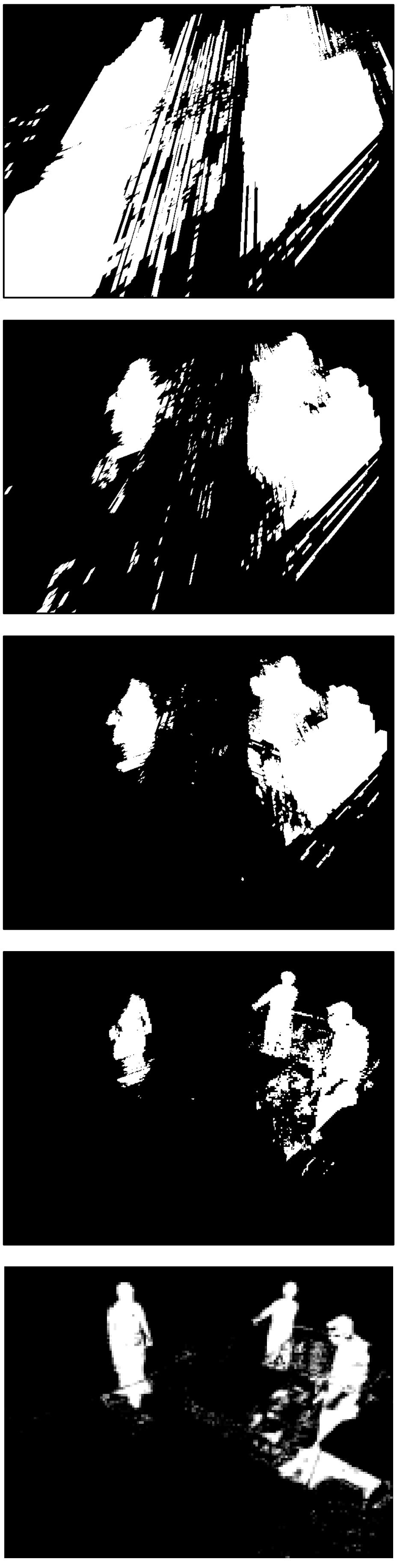

Figure 17: Results of progressive Shape-from-silhouette reconstruction with increasing number of cameras. From top to bottom: projection of the 3D reconstruction with 2, 3, 4 and 5 cameras. Bottom image: original (noisy) 2D foreground.

References

Boyer, E., Franco, J.-S. (2003) 'A hybrid approach for computing visual hulls of complex objects', in *Computer Vision and Pattern Recognition (CVPR'03)*, vol.1, pp. 695–701.

Broadhurst, A., Drummond, T.W., Cipolla, R. (2001), 'A Probabilistic Framework for Space Carving', *in Proc. 8th International Conference on Computer Vision (ICCV'01)*, vol.1, pp. 388–393.

Cheung, G., Kanade, T., Bouguet, J.-Y. & Holler, M. (2000), 'A real time system for robust 3d voxel reconstruction of human motions', *in Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR'00)*, Vol. 2, Hilton Head Island, South Carolina (US), , pp. 714–720.

Erol, A., Bebis, G., Boyle, R. D., Nicolescu, M. (2005), 'Visual Hull Construction Using Adaptive Sampling', in *7th IEEE Workshops on Application of Computer Vision (WACV'05)*, vol.1.

Garcia, O. & Casas, J. (2005), 'Functionalities for mapping 2D images and 3D world objects in a multi-camera environment', *in Proc. 6th Workshop on Image Analysis for Multimedia Interactive Services (WIAMIS'05)*, Montreux, Switzerland, pp. 241–249.

Hartley, R. & Zisserman, A. (2000), *Multiple view geometry in computer vision*, Cambridge Univ.Press.

Kaufman, A., Cohen, D., & Yagel, R. (1993) 'Volume Graphics'. *IEEE Computer* **26** (7), pp. 51–64.

Kutulakos, K.N., Seitz, S.M (2000), 'A Theory of Shape by Space Carving', *International Journal of Computer Vision* **38** (3), pp. 199–218.

Landabaso, J. & Pardas, M. (2005), 'Foreground regions extraction and characterization towards real-time object tracking', *in Proc. Joint Workshop on Multimodal Interaction and Related Machine Learning Algorithms (MLMI'05)*, Edinburgh, UK.

Lu, T. (1996), 'The enumeration of trees with and without given limbs', *Discrete Math*, 154 (1-3), pp. 153–165.

Ma, Y., Soatto, S., Kosecka, J. & Shankar~Sastry, S. (2003), *An Invitation to 3-D Vision: From Images to Geometric Models*, Springer Verlag.

Matusik, W., Buehler, C., McMillan, L. (2001), 'Polyhedral visual hulls for real-time rendering', *in Proc. 12th Eurographics Workshop on Rendering* EGWR'01, London.

Matusik, W., Buehler, C., Raskar, R., Gortler, S.~J. & McMillan, L. (2000), 'Image-based visual hulls', *in Proc. 27th conf. on Computer graphics & interactive techniques (SIGGRAPH'00)*, pp. 369–374.

Puech, W., Bors, A.G., Pitas, I., Chasssery, J.-M. (2001) 'Projection Distortion Analysis for Flattened Image Mosaicing from Straight Uniform Generalized Cylinders', *Pattern Recognition* **34** (8), pp. 1657–1670.

Stauffer, C. & Grimson, W. (2000), 'Learning patterns of activity using real-time tracking', *IEEE Trans. on PAMI* **22** (8), pp. 747–757.

Zhang, Z. (1998), 'Determining the Epipolar Geometry and its Uncertainty: A Review', *International Journal of Computer Vision* **27** (2), pp. 161–198.