A 3D Image Mosaicing System Using the Factorization Method

César Branco
Instituto de Sistemas e Robótica *
Instituto Superior Técnico
Av. Rovisco Pais 1096, Lisboa, Portugal

João Costeira
Instituto de Sistemas e Robótica
Instituto Superior Técnico
Av Rovisco Pais, 1096 Lisboa, Portugal
jpc@isr.ist.utl.pt

Abstract - **Recovering the 3D structure of a dynamic scene from an image sequence is one of the most exciting and extensively studied issues in Computer Vision. We present a shape recovery system based on the factorization method which is able to determine the 3D shape of selected feature points. The algorithm is able to maintain shape representation in a recursive way even when parts of the objects are occluded or leave the field of view. The research documented here is part of the effort that IST is devoting to building a 3D system based on sequences of images.**

1 Introduction

Since the early days of computers, the scientific community has been working towards providing robots with sensing capabilities. Vision is the most promising and yet the most challenging of the artificial senses. From images, biological systems collect most of the information they need to manipulate, navigate and interact with the environment, suggesting that visual sensors are fundamental to providing artificial systems with perception capabilities.

Recovering the 3D structure of a dynamic scene from an image sequence is one of the most extensively studied problems in Computer Vision. This is because not only do objects and robots themselves move, but also motion provides important cues about 3D geometry.

Our aim is to be able to build incremental 3D panoramic images (also known as image mosaics) of the observed environment so that robots can navigate and reason about the world they move in. The usefulness of image mosaics was made obvious to the public by the recent Mars exploration endeavor, where large panoramic images where made by integrating smaller images captured by the camera in different viewing positions.

Admitting that a set of image feature points have been identified and tracked in an image sequence, we aim at building 3D maps with geometric structure and texture represented explicitely. Admitting that we are able to select and track features, the main issues involved in doing this are the following:

- How can 3D shape be computed with images captured by a moving camera.
- How can the 3D shape representation be recursively updated when features disappear from the field of view or become occluded.
- Given a real time algorithm where only a few data is available, how can new features that come into the field of view be inserted into the representation.
- How do we map texture on the object's shape.

*Currently at Laboratório de Termombustão-IST

This papers deals with the last three issues outlined above. We use the recursive factorization method [TK92, MK94] as a framework to compute 3D shape in a recursive fashion. The method produces reliable estimates of shape and motion given a sequence of feature matches.

2 The Factorization Method Revisited

The core of this method is a precedure based on singular value decomposition that separates a matrix of measurements into the product of two matrices which represent, respectively, the shape and motion of an object.

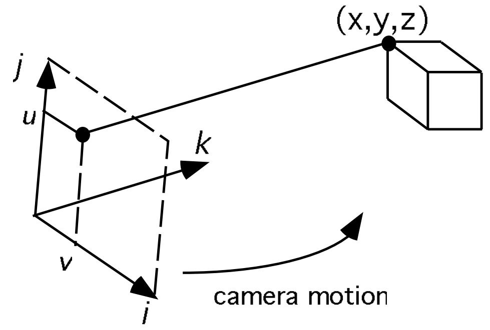


Figure 1: The camera and the object with its coordinate system

Assuming the orthographic projection model and the coordinate systems of figure 1, where i_f j_f k_f represent the camera unit vectors (motion), the image coordinates (u_{fi}, v_{fi}) of the image point i at frame f and its 3D coordinates is given by the following coordinate transform

$$u_{fi} = \begin{bmatrix} i_{x_f} & i_{y_f} & i_{z_f} \\ j_{x_f} & j_{y_f} & j_{z_f} \end{bmatrix} \begin{bmatrix} S_{x_i} \\ S_{y_i} \\ S_{z_i} \end{bmatrix} + \begin{bmatrix} t_{x_f} \\ t_{y_f} \end{bmatrix} \qquad (1)$$

where t_{x_f}, t_{y_f} represent the translation component and $S_{x_i}, S_{y_i}, S_{z_i}$ the 3D position of point i. If we tracked N points in F frames, all measurements can be collected in a single matrix $\hat{\mathbf{W}}$ with the following structure:

$$\hat{\mathbf{W}} = \begin{bmatrix} u_{11} & \dots & u_{1N} \\ \vdots & & \vdots \\ u_{F1} & \dots & u_{FN} \\ v_{11} & \dots & v_{1N} \\ \vdots & & \vdots \\ v_{F1} \dots & v_{FN} & \end{bmatrix} \quad (2)$$

$$\hat{\mathbf{W}} = \overbrace{\left[\begin{array}{c|c} i_{x_1} i_{y_1} i_{z_1} & t_{x_1} \\ \vdots & \vdots \\ i_{x_F} i_{y_F} i_{z_F} & t_{x_F} \\ j_{x_1} j_{y_1} j_{z_1} & t_{y_1} \\ \vdots & \vdots \\ j_{x_F} j_{y_F} j_{z_F} & t_{y_F} \end{array}\right]}^{(camera\ motion)} \overbrace{\begin{bmatrix} s_{x_1} & & s_{x_N} \\ s_{y_1} & \dots & s_{y_N} \\ s_{z_1} & & s_{z_N} \\ 1 & & 1 \end{bmatrix}}^{(object\ shape)} . \quad (3)$$

$$\hat{\mathbf{W}} = [\mathbf{M}|T] \begin{bmatrix} \mathbf{S} \\ \underline{1} \end{bmatrix} \quad (4)$$

The world coordinate system is arbitrary, therefore it can be set at the object's center of mass. Then, the translation components (t_{x_f}, t_{y_f}) can be determined by computing the centroid of the image features (because $\Sigma \mathbf{S}_i = \underline{0}$). Subtracting it to matrix $\hat{\mathbf{W}}$[1] we obtain the final normalized motion equations

$$\mathbf{W} = \mathbf{MS}. \quad (5)$$

Recovering shape and motion is equivalent to factorizing matrix $\mathbf{W}$ into a product of two rank 3 matrices, one representing motion and the other representing shape. Singular value decomposition (SVD)[GVL89] is the best method to estimate both matrices in the least square sense. So, $\mathbf{W}$ can be expressed as

$$\mathbf{W} = \mathbf{MS} = \mathbf{U}\Sigma\mathbf{V}^T, \quad (6)$$

where $\mathbf{U}(2F \times 3)$ and $\mathbf{V}^T(3 \times N)$ are two orthonormal matrices and Σ is a 3×3 diagonal matrix. Since the decomposition is not unique we must determine an affine transformation $\mathbf{A}$ such that

$$\mathbf{W} = \mathbf{UAA}^{-1}\Sigma\mathbf{V}^T \quad (7)$$
$$\mathbf{M} = \mathbf{UA} \quad (8)$$
$$\mathbf{S} = \mathbf{A}^{-1}\Sigma\mathbf{V}^T \quad (9)$$

This can be done imposing rigidity constraints. In other words, the rows of matrix $\mathbf{M}$ which represent the unit vectors of the camera coordinate system must satisfy the constraints (see eq 3):

$$i_f^T i_f = 1 \quad i_f^T j_f = 0 \quad j_f^T j_f = 1 \quad [f \in (1 \dots F)]. \quad (10)$$

[1] we could also choose any point as the origin of the coordinate system and then subtract its image coordinates to all other points. From now on we assume this operation is always performed prior to any shape computation

Expressing the above equation relative to the decomposition 7 we obtain the set of equations

$$\mathbf{U}_f A^T A \mathbf{U}_f = 1 \quad \mathbf{U}_f A^T A \mathbf{U}_{f+F} = 0$$
$$\mathbf{U}_{f+F} A^T A \mathbf{U}_{f+F} = 1 \qquad [f = (1 \dots F)] \quad (11)$$

where $\mathbf{U}_f$ represents row f of matrix $\mathbf{U}$. Noting that $\mathbf{L} = \mathbf{A}^T\mathbf{A}$ is a 3×3 symetric matrix, the constraints (11) form a system of 3 linear equations on the 6 unknown elements of L, say $(l_1, \dots, l_6)$:

$$\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} g(U_f, U_f) \\ g(U_f, U_{f+F}) \\ g(U_{f+F}, U_{f+F}) \end{bmatrix} \begin{bmatrix} l_1 \\ l_2 \\ l_3 \\ l_4 \\ l_5 \\ l_6 \end{bmatrix} \quad (12)$$

where $g()$ is a 1×6 vector of coefficients.

Since we have F sets of constraints (12) we can compute the least square error estimate of the elements of L from an overdetermined set of $3F$ equations

$$\begin{bmatrix} 1 \\ 0 \\ 1 \\ \vdots \\ 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} g(U_1, U_1) \\ g(U_1, U_{1+F}) \\ g(U_{1+F}, U_{1+F}) \\ \vdots \\ g(U_F, U_F) \\ g(U_F, U_{2F}) \\ g(U_{2F}, U_{2F}) \end{bmatrix} \begin{bmatrix} l_1 \\ l_2 \\ l_3 \\ l_4 \\ l_5 \\ l_6 \end{bmatrix} \quad (13)$$

Using again SVD, matrix L can be easily factorized into matrices $\mathbf{A}^T\mathbf{A}$ (see [TK92] for further details). Motion can be computed by $\mathbf{M} = \mathbf{UA}$ and shape by $\mathbf{S} = \mathbf{A}^{-1}\Sigma\mathbf{V}^T$.

Recursive Factorization

The algorithm outlined above is not practical if one intends to process data in real time since all measurements must be available in order to compute the decomposition. In this case, storage requirements grow indefinitely (the number of rows in $\mathbf{W}$increases with the number of frames) and the estimate is produced only at the last time instant. Morita and Kanade [MK94] modified this method so that storing old data is no longer required to estimate motion and shape. In brief, the method relyes on the fact that matrix $\mathbf{V}$ is the solution of the eigensystem $\mathbf{V}\Sigma = (\mathbf{W}^T\mathbf{W})\mathbf{V}$, which are constant size matrices. At frame f, when new data is acquired, the measurement matrix $Z_f = \mathbf{W}_f^T\mathbf{W} = Z_{f-1} + u_f u_f^T + v_f v_v^T$ (u_f, v_f are the vectors with the X and Y coordinates of the image points) are is updated and a new stable matrix $\mathbf{V}_f$ is computed at each frame. Noting that $\mathbf{U}_f = u_f^T \mathbf{V}_f \Sigma{-}1_f, \mathbf{U}_{f+1} = v_f^T \mathbf{V}_f \Sigma{-}1_f$ and that 13 is already a recursive equation, matrices $\mathbf{\Lambda}_f$ and $\mathbf{A}_f$ can also be computed recursively and thus, shape and motion can also be computed at frame f.

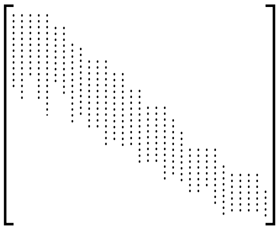
In summary, at each frame the recursive factorization produces the following matrices:

- $\mathbf{V}_f$ - base of the subspace spanned by the rows of $\mathbf{W}_f$

- $\mathbf{A}_f$ - matrix resulting from the rigidity constraints
- $i_f = [i_{f_x} i_{f_y} i_{f_z}]$ and $i_f = [i_{f_x} i_{f_y} i_{f_z}]$ the motion matrix ($k_f = i_f \times j_f$).
- $\mathbf{S}_f$ - Estimate of the object shape at frame f.

3 Handling Occlusion and New Features

In real situations, feature points are visible for a limited time since they go into and out of the field of view as the camera moves (imagine a camera going around one object). To illustrate the situation we show in figure 2 the typical aspect of the measurements in matrix $\mathbf{W}$, where the dots in the figure represent available measurements. The dots in each column represent the time slice during which the point is visible.



a)

Figure 2: Measurements matrix with missing data

Since we use a recursive approach at each frame we only have available one row of data. We will show that if a point is not visible in subsequent frames its future position can be estimated from visible points by exploiting linear dependence relations. However, this mechanism does not work backwards, that is, if a new point is selected we will not be able to reconstruct the past trajectory. The algorithm is recursive so past measurements are lost and the backward estimation process lacks data.

Occlusion

Let us admit that the tracking mechanism can detect either that a given feature became occluded or it is unreliable and thus eliminated. The factorization couldn't proceed since matrix Z_f would have missing data. Let's admit that from the N fetures only M of them are valid. Recall that from the recursive factorization we have matrix $\mathbf{V}_{f-1}$ which is a base for the rows of $\mathbf{W}$. Permuting columns accordingly, the measurements can be expressed as a linear combination of the rows of $\mathbf{V}^T$ and the translation vector in the following way

$$\left[\begin{array}{c|c} u_{f1} \cdots u_{fM} & u_{fM+1} \cdots u_{fN} \\ v_{f1} \cdots v_{fM} & v_{fM+1} \cdots v_{fN} \end{array}\right] = \begin{bmatrix} \alpha & \beta & \gamma \\ \sigma & \delta & \epsilon \end{bmatrix} \begin{bmatrix} \mathbf{V}^T_{f-1(:,1:M)} & \mathbf{V}^T_{f-1(:,M+1:N)} \end{bmatrix} + \begin{bmatrix} t_{x_f} \cdots t_{x_f} \\ t_{y_f} \cdots t_{y_f} \end{bmatrix}$$

$$[\mathbf{W}_1 | \mathbf{W}_2] = \begin{bmatrix} \alpha & \beta & \gamma & t_{x_f} \\ \sigma & \delta & \epsilon & t_{y_f} \end{bmatrix} \begin{bmatrix} \mathbf{V}^T_{f-1(:,1:M)} & \mathbf{V}^T_{f-1(:,M+1:N)} \\ 1 \cdots 1 & 1 \cdots 1 \end{bmatrix} = \begin{bmatrix} \alpha & \beta & \gamma & t_{x_f} \\ \sigma & \delta & \epsilon & t_{y_f} \end{bmatrix} [V_1^T | V_2^T]$$

where $\mathbf{W}_1$ and $\mathbf{V}_1^T$ are the submatrices formed with the columns corresondent to available data, $\mathbf{W}_2$ is the matrix of the unknown measurements and $\mathbf{V}_2^T$ the correspondent columns of the base $\mathbf{V}_{f-1}$. The unknown coefficients $(\alpha, \beta, \gamma, \sigma, \delta, \epsilon)$ and the translation vector (centroid) can be estimated in the least square sense from the available measurements by:

$$\begin{bmatrix} \alpha & \beta & \gamma & t_{x_f} \\ \sigma & \delta & \epsilon & t_{y_f} \end{bmatrix} = \mathbf{W}_1 \mathbf{V}_1 \left(\mathbf{V}_1^T \mathbf{V}_1\right)^{-1} \quad (14)$$

provided that the rank of $\mathbf{V}_{f-1}$ is 3. Knowing the coefficients, the missing measurements can be predicted reconstructing $\mathbf{W}_2$ as

$$\mathbf{W}_2 = \begin{bmatrix} \alpha & \beta & \gamma & t_{x_f} \\ \sigma & \delta & \epsilon & t_{y_f} \end{bmatrix} \mathbf{V}_2. \quad (15)$$

At this point the factorization can be updated.

Including new features

In real time operation, the feature tracker not only tracks features but it is also selecting new trackable features all the time. During the tracking process a reliability measure is computed and if the feature position error reaches a certain threshold it is abandoned. New features are then selected and tracked. Also as the camera moves, say with constant translational velocity, features tend to leave the field of view and new ones come in.

Again, the shape update cannot proceed. Since shape is computed after the eigendecomposition of $Z_f = \mathbf{W}^T_{f-1}\mathbf{W}_{f-1} + u_f^T u_f + v_f^T v_f$, if new measurementes are available, vectors u_f and v_f would have its size increased but the corresondent values in $\mathbf{W}^T_{f-1}\mathbf{W}_{f-1}$ would be missing. There is no possibility of repeating the above process backwards. The only straightforward solution would be to preserve motion matrix $\mathbf{M}$ and the translation components which again is not possible since they grow indefenitely. Suppose that we have tracked new features for some time. The measurements matrix would be formed by the submatrices shown in figure 3 where $\mathbf{W}_2$ is unknown.

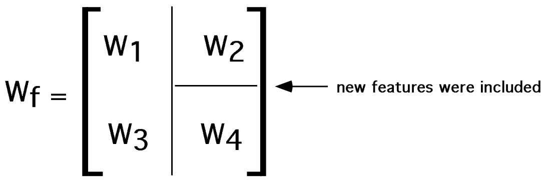


Figure 3: Structure of the observations matrix when new features have been added.

Then,

$$Z_f = \mathbf{W}_f^T \mathbf{W}_f$$

$$Z_f = \begin{bmatrix} \mathbf{W}_1^T\mathbf{W}_1 + \mathbf{W}_3^T\mathbf{W}_3 & \mathbf{W}_1^T\mathbf{W}_2 + \mathbf{W}_3^T\mathbf{W}_4 \\ \mathbf{W}_2^T\mathbf{W}_1 + \mathbf{W}_1^T\mathbf{W}_3 & \mathbf{W}_2^T\mathbf{W}_2 + \mathbf{W}_4^T\mathbf{W}_4 \end{bmatrix} \quad (16)$$

Suppose we collected all motion estimates in a matrix, say $\mathbf{M}_{old}$, since the beginning up to the instant when the new features started being tracked (see figure 3), and the new estimates are also stored in a matrix, say $\mathbf{M}_{new}$. Then the observation equation becomes

$$\begin{bmatrix} \mathbf{W}_1 & \mathbf{W}_2 \\ \mathbf{W}_3 & \mathbf{W}_4 \end{bmatrix} = \begin{bmatrix} \mathbf{M}_{old} \\ \mathbf{M}_{new} \end{bmatrix} [\mathbf{S}_{old}\mathbf{S}_{new}] \quad (17)$$

Keeping the factorization going with the measurements $\mathbf{W}_3$ we can estimate $\mathbf{M}_{new}$ as if no new features were selected. Then, the shape of the new points can be computed by:

$$\mathbf{S}_{new} = \left(\mathbf{M}_{new}^T\mathbf{M}_{new}\right)^{-1}\mathbf{M}_{new}^T\mathbf{W}_4. \quad (18)$$

Note that this equation can be recursively updated as new features (rows of $\mathbf{W}_4$) are available. With the new shape estimate we can now proceed to compute Z_f. The unknown terms are given by:

$$\begin{aligned} \mathbf{W}_2^T\mathbf{W}_2 &= \mathbf{S}_{new}^T\mathbf{M}_{old}^T\mathbf{M}_{old}\mathbf{S}_{new} && (19) \\ \mathbf{W}_1^T\mathbf{W}_2 &= \mathbf{S}_{old}^T\mathbf{M}_{old}^T\mathbf{M}_{old}\mathbf{S}_{new} && (20) \end{aligned}$$

Here we already have recursive constant size matrices since

$$\begin{aligned} \Omega_f &= \mathbf{M}_{old}^T\mathbf{M}_{old} && (21) \\ \Omega_f &= \Omega_{f-1} + i_f i_f^T + j_f j_f^T && (22) \end{aligned}$$

In practice we monitor the singular values of $\mathbf{W}_4$ and when the third reaches a certain threshold we proceed with the computations. The consequence is that $\mathbf{W}_4 and \mathbf{W}_3$ must overlap for at least 3 frames.

With the full shape matrix available $\mathbf{S}_f = [\mathbf{S}_{old}\mathbf{S}_{new}$ the new row base $\mathbf{V}_f$ is computed using the QR factorization as in [MK94].

4 Experimental Results

The algorithm was tested in several conditions. Features were selected and tracked (the tracking and feature selection procedure is described in detail in [CK97]). Occlusion or unreliability was determined by measuring the error between the matched patterns in consecutive images. Due to lack of space we do not describe the texture mapping in detail. In summary, we interpolate linearly the 3D feature positions using Delaunay triangulation and then we map the correspondent image texture on the surface. Figure 4 shows two frames of sequence obtained from a camera moving around the cube, and in figure 5 we show the obtained shape reconstruction and feature triangulation.

Note the correctness of the reconstruction. We use some heuristics to update the triangle texture in order to increase resolution. The right side of the cube is viewed in a quite skewed angle. The result is that even though the triangle might have a large area in the

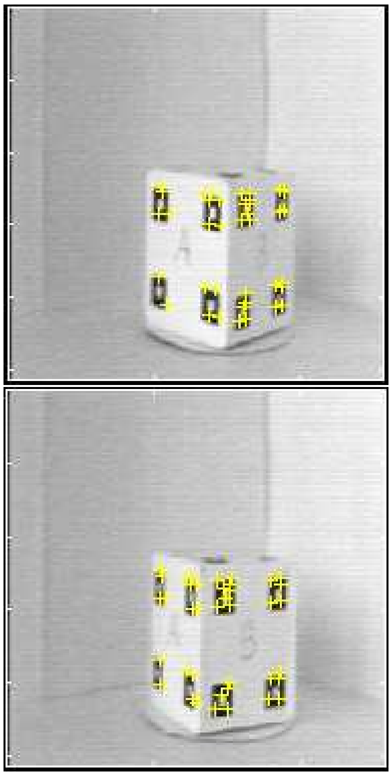


Figure 4: Image sequence of a cube. Two images of the sequence.

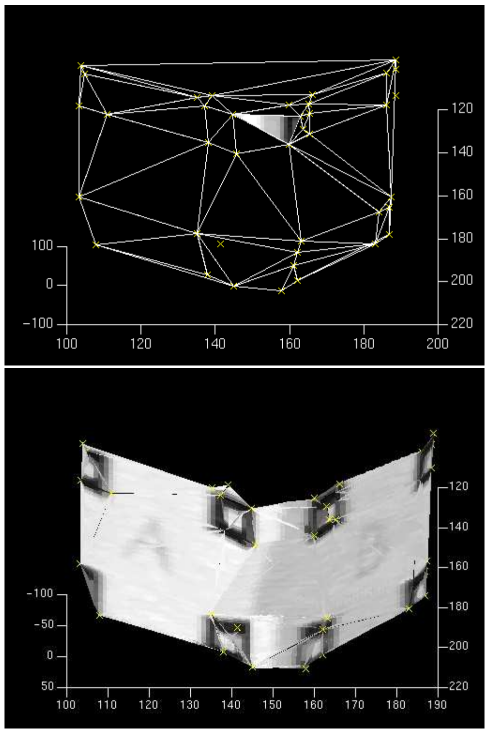


Figure 5: Feature triangulation and texture mapped on plane interpolation

3D representation its texture has a few pixels. However as the camera moves and that particular side becames more parallel to the image plane the 3D texture is remmaped, consequently its detail increases.

In figure 6 we show the effect of occlusions and of new feature inclusion. The sequence is obtained by a camera rotating around a cup. The top picture shows the first image of the sequence with the first selected features superimposed. Many of these features were wrongly detected (for example in the occluding contour) but they will be eliminated by the tracking procedure by monitoring the tracking error (square error between the current and the original pattern). In the mid picture we show the computed shape of the cup as seen from above after a complete turn around it. Finally in the last picture we show a partial texture of the cup showing a fairly good representation of the object.

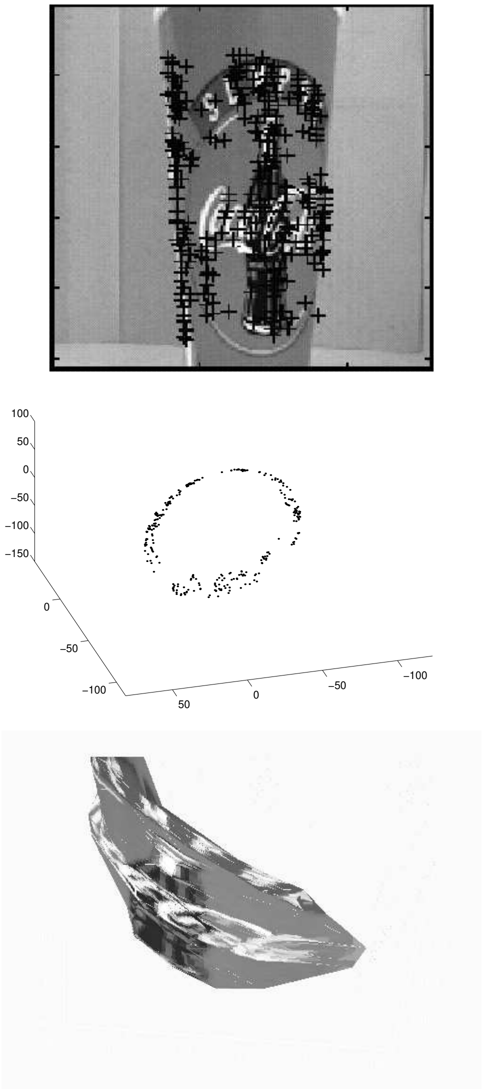


Figure 6: Cup sequence. Top figure shows the first image of the sequence with the tracked features. The middle figure depicts the computed shape seen from above and the bottom image, the interpolated shape

5 Conclusions

Tridimensional shape computation is of paramount importance in mobile robotic systems. Furthermore real time operation impose particular constraints, namely recursivity. In real situations, shape computation algorithms must be able deal with the possibility of a continously moving camera viewpoint. This means that objects that are being viewd now will disappear in the future and new objects will come into the field of view. The ability to dynamically update 3D shape as the scene changes without recurring to past stored measurements is of fundamental importance. We have described a method that adds this capability to the well known (recursive) factorization method for shape and motion determination. This method allows the buildup of 3D image mosaics of the viewed environment.

Future developments include the extension to more complicated features such as lines and parametrized curves in order produce a better reconstruction of the object. Even though we used an orthographic model for the camera, the reformulation with more complex models like paraperspective (see [PK93]) or even projective seems feasible.

References

[CK97] Joao Costeira and Takeo Kanade. A multibody factorization method for independently moving objects:full report. Technical Report CMU-RI-TR-97-30, Robotics Institute, Carnegie Mellon University, July 1997. Also available at http://www.isr.ist.utl.pt/~jpc.

[GVL89] G. Golub and C.F . Van Loan. *Matrix Computations*. The Johns Hopkins University Press, Baltimore, Md, 1989.

[MK94] Toshihiko Morita and Takeo Kanade. A sequential factorization method for recovering shape and motion from image streams. In *ARPA IUS workshop*, 1994.

[PK93] C Poelman and T. Kanade. A paraperspective factorization method for shape and motion recovery. Technical Report CS-93-219, School of Computer Science, Carnegie Mellon University, December 1993.

[TK92] Carlo Tomasi and Takeo Kanade. Shape and motion from image streams under orthography: A factorization method. *International Journal on Computer Vision*, 9(2):137–153, 1992.