REAL TIME $3D$ NAVIGATION IN A STATIC VIRTUALIZED SCENE FROM A LIMITED SET OF $2D$ DATA

Katia Fintzel

SIMULOG, Les taissounières HB2
route des Dolines, BP 277
06905 Sophia Antipolis, France
Katia.Fintzel@esprico.fr

Jean-Luc Dugelay

Institut EURECOM, CICA
2229 route des Crêtes, BP 193
06904 Sophia Antipolis, France
Jean-Luc.Dugelay@eurecom.fr

ABSTRACT

Among the new data processing services and applications, virtual navigation in a real $3D$ world takes more and more importance and requires specific research. With regard to the reactions of the most important part of net surfers for example, we think that for a user of an interactive immersive tool, it is absolutely necessary to obtain ***a real time rendering of the virtual world, as natural and realistic as possible***. That is why we focus our work on an image based approach, avoiding the slow and heavy handling of $3D$ models of the scene, generally not very realistic and typical in geometry based approaches. Our method uses for its part the synthesis of photorealistic views based on the trilinearity theory combined with image mosaicking. It especially minimizes the algebraic processings needed for view synthesis in order to offer in real time a natural rendering of the scene to the user, according to his/her relative movement from his/her initial position.

1. INTRODUCTION

The interest for virtual navigation in static scenes has really incredibly extended, because of the number of potential applications in industrial domains. The process of virtual navigation generally requires a complete $3D$ model of the scene, whose acquisition is a real toil, and must often suffers from a serious lack of realism. That is why we focus on virtualized navigation in a real scene, modelled only by a few $2D$ uncalibrated photographs rather than a $3D$ CAD representation.

Virtualized navigation aims at offering a potential user his/her current global point of view of the scene, according to his/her initial position and relative movement from this position. Ideally we would like to perform the synthesis of the $2D$ point of view of the user, without reconstructing the $3D$ model of the scene and obviously keeping the calibration stage of the video camera system implicit. In this context, we use the trilinearity theory, which allows to synthesize each unknown point of view of a scene from two neighboring views as recalled in section 2.1. In section 2.2, we briefly present a new approach using underlayers of the synthesized view, developed in the context of the navigation simulation in virtualized scenes (section 2.3) to increase the visual realism of the $2D$ generated views and ensure a good degree of motion fluidity. The essence of our recent work, as presented in section 3, is the optimization of the algebraic processings needed for the constant update of the underlayers, in order to achieve the real time simulation of virtualized navigation. And finally, we sum up the benefits of our approach and its potential domains of application in the concluding section.

2. VIRTUALIZED NAVIGATION IN A REAL SCENE: PREVIOUS WORK

2.1. Trilinearity to Synthesize Points of View

We propose an algorithm for view synthesis from uncalibrated $2D$ views of a real $3D$ scene based on the trilinear tensors [1], extending the stereovision concepts [2, 3] on three different perspective views of the same scene. If we consider a triplet of views ($i-1$, i and $i+1$) extracted from n original views of the $3D$ scene, a new point of view i' can therefore be generated from its two neighboring initial ordinary images $i-1$ and $i+1$, without any explicit calibration stage, in two steps:

- Analysis: Using seven or more corresponding points in three original uncalibrated views denoted $i-1$, i and $i+1$ (modelled in a discrete approach by the reference texture $i+1$ mapped on three meshes m_{i-1}, m_i and m_{i+1} [4]), eighteen trilinear parameters are estimated [1, 5].
- Synthesis: An unknown intermediate view i' is generated, using all the corresponding mesh nodes of the images $i-1$ and $i+1$ (i.e m_{i-1} and m_{i+1}) and the estimated parameters, algebraically manipulated to simulate a change of the focal length or a $3D$ displacement of the virtual camera relative to the point of view i [6]. Only a synthesis step is required to simulate a change of the current point of view of the considered user, whereas the analysis step remains unchanged.

2.2. Underlayers to Ensure Visual Realism

The coverage of the synthesized view i' depends on the size of the common area of the three initial meshes m_{i-1}, m_i and m_{i+1}, which can be very limited. As a solution to this problem, we introduce a processing based on image mosaicking [7, 8] into the synthesis method.

2.2.1. Homographic Transform

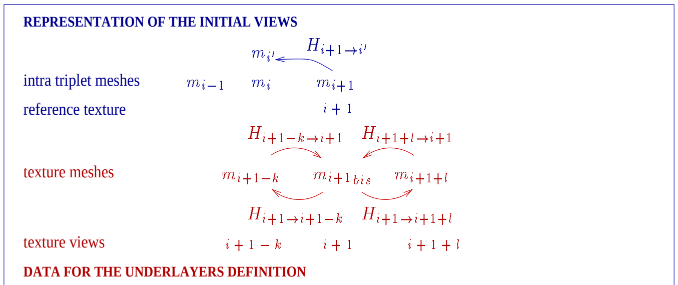
Let us consider two of the three original images for example i and $i+1$: we can relate these views by a geometrical transform called ***homography*** [2, 9] and defined by the 3×3 matrix $H_{i+1\to i}$ $|\boldsymbol{p_i} = H_{i+1\to i}.\boldsymbol{p_{i+1}}$ (with $\boldsymbol{p_i}$ and $\boldsymbol{p_{i+1}}$ the mesh nodes of the respective images i and $i+1$). Note that, in general $\boldsymbol{H_{i+1\to i}}$

only performs a $2D$ approximation between the two neighboring images under consideration, except in the particular case of pure rotations using a single camera (between the two different points of view). $\boldsymbol{H_{i+1\rightarrow i}}$ is then the expression of the $3D$ motion of the video camera from the position corresponding to the point of view $i+1$ to the position corresponding to i.

2.2.2. Estimation of Underlayers

Using homographic transforms, an image mosaicking approach is used to reduce the effect of the limited covering area, superimposing three images (figure 1 steps s_2, s_4 and s_5) to render a realistic point of view of the scene:

- The usual ***intra triplet*** synthesized view i' displayed in front of the two other views and really simulating a $3D$ movement of the video camera;
- And two approximations, denoted ***underlayers*** obtained by image mosaicking between i' and respectively two other relevant views compared to the reference texture $i+1$ (a preceding $i+1-k$ and a following one $i+1+l$ in terms of direction of the motion and called ***texture views*** for example see figure 3).



ALGORITHM:

```
1. Trilinear parameters estimation for the synthesis of $m_i$ from $m_{i-1}$ and $m_{i+1}$
2. Estimation of $H_{i+1-k\rightarrow i+1} | H_{i+1-k\rightarrow i+1}(m_{i+1-k}) = m_{i+1\,bis}$
3. Deduction of $H_{i+1\rightarrow i+1-k} | H_{i+1\rightarrow i+1-k}(m_{i+1\,bis}) = m_{i+1-k}$
4. Estimation of $H_{i+1+l\rightarrow i+1} | H_{i+1+l\rightarrow i+1}(m_{i+1+l}) = m_{i+1\,bis}$
5. Deduction of $H_{i+1\rightarrow i+1+l} | H_{i+1\rightarrow i+1+l}(m_{i+1\,bis}) = m_{i+1+l}$
4. Add $n$ nodes to texture mesh $m_{i+1-k} \longrightarrow m^{\star}_{i+1-k}$
5. $H_{i+1\rightarrow i+1-k}(m_{i+1}) \oplus m^{\star}_{i+1-k} \longrightarrow m^{\star\star}_{i+1-k}$
7. Add $n$ nodes to texture mesh $m_{i+1+l} \longrightarrow m^{\star}_{i+1+l}$
8. $H_{i+1\rightarrow i+1+l}(m_{i+1}) \oplus m^{\star}_{i+1+l} \longrightarrow m^{\star\star}_{i+1+l}$
    For each video camera motion :
        $s_1$  Trilinear parameters alteration
        $s_2$  $m_{i-1}, m_{i+1}$ and altered parameters $\longrightarrow m_{i'}$        virtualized view   $i'$
        $s_3$  Estimation of $H_{i+1\rightarrow i'}$
        $s_4$  $H_{i+1\rightarrow i'}(H_{i+1-k\rightarrow i+1}(m^{\star\star}_{i+1-k})) = m_{i'-k}$        first underlayer   $i'-k$
        $s_5$  $H_{i+1\rightarrow i'}(H_{i+1+l\rightarrow i+1}(m^{\star\star}_{i+1+l})) = m_{i'+l}$        second underlayer   $i'+l$
```

Figure 1: Definition of the underlayers of a virtualized view using homographic transforms between the inter triplet images and the current virtualized view.

2.3. Navigation in a Real Scene Using Virtualized Points of View

In order to simulate the visit of a virtualized scene, we must generate enough different images of the environment from different triplets of initial views and link these resulting syntheses together. The user's eyes are then considered as a virtual camera, whose positions and motions allow us to continually synthesize his/her coherent point of view in the scene. Let us consider several triplets of images, we are able to simulate motions around each triplet by altering the trilinear parameters used to build the synthesis i' from $i-1$ and $i+1$ [4]. Using consecutive triplets of views, we can propagate the same type of motion from one triplet ($i-1$, i, $i+1$) to the next ($i+2$, $i+3$, $i+4$) in terms of the direction of the movement, testing each time the credibility of the synthesized views i' for the triplet ($i-1$, i, $i+1$) or $i'+3$ for ($i+2$, $i+3$, $i+4$) [10]. Examples of such Mpeg encoded sequences can be found at *http://www.eurecom.fr/˜image/spatialisation.html*. These travelling simulations are fair but visually uncomfortable for the user, because of the visual artefacts introduced by the ***triplet transitions*** between the initial tri-views sequences ($i-1$, i, $i+1$) and ($i+2$, $i+3$, $i+4$) [10]. However these artefacts are considerably reduced by the underlayers method (figure 2).

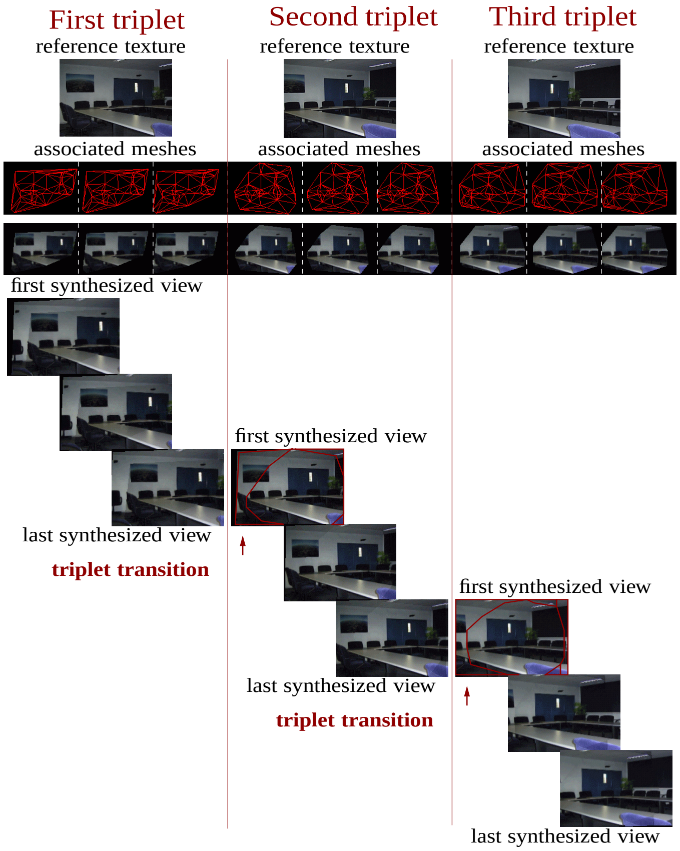


Figure 2: Simulation of virtualized navigation in a real scene from three triplets of views.

3. REAL TIME NAVIGATION

3.1. How to Achieve Real Time Performance?

We have shown that the use of homographic transforms for the underlayers definition can secure the visual realism of the virtual navigation process. In this way, we currently offer the user a quasi-complete point of view of the scene from his/her virtual position. But this specification requires the estimation of the synthetic representation of the user's point of view of the scene and its underlayers at each of his/her movements, requiring an algebraic estimation and two compositions of homographic transforms (section 2.2.2 figure 1 steps s_3, s_4 and s_5). To minimize the processing time between two successive displayed points of view rendered to the user, depending of his/her pose and motion, we modify these expensive computations as follows.

3.2. Optimization of the Underlayers Estimation

3.2.1. *Composition of Homographic Transforms*

With $i+1-k$ and $i+1+l$, we can increase any synthesized unknown point of view i' mapping two underlayers $i'-k$ and $i'+l$, obtained by homographic transforms [10] (section 2.2.2). But this implies, at each movement of the user, the estimation of an homographic transform $\boldsymbol{H_{i+1\rightarrow i'}}$ between $i+1$ and the virtualized point of view i' (figure 1 step s_3), and its composition with the homographic transforms $\boldsymbol{H_{i+1-k\rightarrow i+1}}$ from $i+1-k$ to $i+1$ first and $\boldsymbol{H_{i+1+l\rightarrow i+1}}$ from $i+1+l$ to $i+1$ (figure 1 steps s_4 and s_5).

In order to skip this costly process, at each movement of the user, we propose to directly combine the initial transforms $\boldsymbol{H_{i+1-k\rightarrow i+1}} \circ \boldsymbol{H_{i+1\rightarrow i}}$ and $\boldsymbol{H_{i+1+l\rightarrow i+1}} \circ \boldsymbol{H_{i+1\rightarrow i}}$ with $[R|t]$ the relative movement of the user continuously known and updated (In short $\boldsymbol{[R|t]}$ defines the virtual displacement between i and i') . By this way, we considerably reduce the algebraic processings.

More precisely: $\boldsymbol{p_{i-1}}, \boldsymbol{p_i}$ and $\boldsymbol{p_{i+1}}$ are the m_{i-1}, m_i and m_{i+1} nodes defined from the ***intra triplet*** views $i-1$, i and $i+1$. As detailed in 2.2.1:

$$\exists \boldsymbol{H_{i+1\rightarrow i}} | \boldsymbol{p_i} = \boldsymbol{H_{i+1\rightarrow i}} . \boldsymbol{p_{i+1}} \qquad (1)$$

In the same way, $\boldsymbol{p_{i+1-k}}, \boldsymbol{p_{i+1_{bis}}}$ and $\boldsymbol{p_{i+1+l}}$ are the ***texture meshes*** m_{i+1-k}, $m_{i+1_{bis}}$ $(m_{i+1_{bis}} \neq m_{i+1})$ and m_{i+1+l} nodes from the ***inter triplet*** views $i+1-k$, $i+1$ and $i+1+l$:

$$\left\{ \begin{array}{l} \exists \boldsymbol{H_{i+1-k\rightarrow i+1}} | \boldsymbol{p_{i+1_{bis}}} = \boldsymbol{H_{i+1-k\rightarrow i+1}} . \boldsymbol{p_{i+1-k}} \\ \exists \boldsymbol{H_{i+1+l\rightarrow i+1}} | \boldsymbol{p_{i+1_{bis}}} = \boldsymbol{H_{i+1+l\rightarrow i+1}} . \boldsymbol{p_{i+1+l}} \end{array} \right.$$

We recall here that m_{i+1} is the intra triplet mesh built from the images $i-1$, i and $i+1$ and $m_{i+1_{bis}}$ is the texture mesh built from the images $i+1-k$, $i+1$ and $i+1+l$. We estimate once and for all the homographic transforms to define the underlayers of the optical system in its initial configuration (corresponding to the original position of the user). We add to m_{i+1-k} and m_{i+1+l} new nodes from non investigated areas of $i+1-k$ and $i+1+l$ to obtain the meshes $m^{\star}_{i+1-k}$ and $m^{\star}_{i+1+l}$, and homologous points to the mesh nodes m_{i+1}, to obtain the meshes $m^{\star\star}_{i-1-k}$ and $m^{\star\star}_{i+1+l}$ (figure 1 steps 4 to 8):

$$\begin{array}{l} \boldsymbol{H_{i+1\rightarrow i+1-k}}(m_{i+1}) \oplus m^{\star}_{i+1-k} = m^{\star\star}_{i+1-k} \\ \boldsymbol{H_{i+1\rightarrow i+1+l}}(m_{i+1}) \oplus m^{\star}_{i+1+l} = m^{\star\star}_{i+1+l} \end{array}$$

From which:

$$\left\{ \begin{array}{ll} \forall \boldsymbol{p_{i+1}} \in m_{i+1} & \exists \boldsymbol{p_{i+1-k}} \in m^{\star\star}_{i+1-k} \\ & | \boldsymbol{p_{i+1}} = \boldsymbol{H_{i+1-k\rightarrow i+1}} . \boldsymbol{p_{i+1-k}} \\ \forall \boldsymbol{p_{i+1_{bis}}} \in m_{i+1_{bis}} & \exists \boldsymbol{p_{i+1-k}} \in m^{\star\star}_{i+1-k} \\ & | \boldsymbol{p_{i+1_{bis}}} = \boldsymbol{H_{i+1-k\rightarrow i+1}} . \boldsymbol{p_{i+1-k}} \end{array} \right.$$

$$\left\{ \begin{array}{ll} \forall \boldsymbol{p_{i+1}} \in m_{i+1} & \exists \boldsymbol{p_{i+1+l}} \in m^{\star\star}_{i+1+l} \\ & | \boldsymbol{p_{i+1}} = \boldsymbol{H_{i+1+l\rightarrow i+1}} . \boldsymbol{p_{i+1+l}} \\ \forall \boldsymbol{p_{i+1_{bis}}} \in m_{i+1_{bis}} & \exists \boldsymbol{p_{i+1+l}} \in m^{\star\star}_{i+1+l} \\ & | \boldsymbol{p_{i+1_{bis}}} = \boldsymbol{H_{i+1+l\rightarrow i+1}} . \boldsymbol{p_{i+1+l}} \end{array} \right. \qquad (2)$$

For simplification of the notations, we assume $\{\boldsymbol{p_{i+1}}\} \oplus \{\boldsymbol{p_{i+1_{bis}}}\} \Longrightarrow \{\boldsymbol{p_{i+1}}\}$. Combining the equation 1 with each system of the equation 2, we obtain $\boldsymbol{p_{i'-k}}$ and $\boldsymbol{p_{i'+l}}$ the mesh nodes of $m_{i'-k}$ and $m_{i'+l}$:

$$\boxed{\left\{ \begin{array}{l} \boldsymbol{p_{i'-k}} = \boldsymbol{H_{i+1\rightarrow i}} . \boldsymbol{p_{i+1}} = \boldsymbol{H_{i+1\rightarrow i}} . \boldsymbol{H_{i+1-k\rightarrow i+1}} . \boldsymbol{p_{i+1-k}} \\ \boldsymbol{p_{i'+l}} = \boldsymbol{H_{i+1\rightarrow i}} . \boldsymbol{p_{i+1}} = \boldsymbol{H_{i+1\rightarrow i}} . \boldsymbol{H_{i+1+l\rightarrow i+1}} . \boldsymbol{p_{i+1+l}} \end{array} \right.} \qquad (3)$$

3.2.2. *Real Time Estimation of Underlayers*

If we consider the perspective projection [2] from $3D$ points $\boldsymbol{P}$ to the view i, we have: $\boldsymbol{p_i} = \boldsymbol{M^i}[\boldsymbol{R^i}|\boldsymbol{t^i}].\boldsymbol{P}$. We change the focal length of the video camera ($\boldsymbol{M^i} \Longrightarrow \boldsymbol{M^i_{def}}$) and its relative $3D$ position compared to the view i to simulate an user motion and we have: $\boldsymbol{p_{i'}} = \boldsymbol{M^i_{def}}[\boldsymbol{R}|\boldsymbol{t}][\boldsymbol{R^i}|\boldsymbol{t^i}].\boldsymbol{P}$. Developing and solving this equation, we obtain:

$$\left\{ \begin{array}{lcl} x_{i'} & = & \dfrac{kf^i[r_{11}x_i + r_{12}y_i + r_{13} + \frac{f^i.t_1}{r^i_{31}X + r^i_{32}Y + r^i_{33}Z + t^i_3}]}{r_{31}x_i + r_{32}y_i + r_{33} + \frac{f^i.t_3}{r^i_{31}X + r^i_{32}Y + r^i_{33}Z + t^i_3}} \\ y_{i'} & = & \dfrac{kf^i[r_{21}x_i + r_{22}y_i + r_{23} + \frac{f^i.t_2}{r^i_{31}X + r^i_{32}Y + r^i_{33}Z + t^i_3}]}{r_{31}x_i + r_{32}y_i + r_{33} + \frac{f^i.t_3}{r^i_{31}X + r^i_{32}Y + r^i_{33}Z + t^i_3}} \end{array} \right.$$

And for the underlayers $\boldsymbol{p_{i'-k}}$ and $\boldsymbol{p_{i'+l}}$, given by the equation 2, we derive:

$$\boxed{\begin{array}{l} \left\{ \begin{array}{lcl} x_{i'-k} & = & \dfrac{kf^i[r_{11}x_{i-k} + r_{12}y_{i-k} + r_{13} + \frac{f^i t_1}{s}]}{r_{31}x_{i-k} + r_{32}y_{i-k} + r_{33} + \frac{f^i t_3}{s}} \\ y_{i'-k} & = & \dfrac{kf^i[r_{21}x_{i-k} + r_{22}y_{i-k} + r_{23} + \frac{f^i t_2}{s}]}{r_{31}x_{i-k} + r_{32}y_{i-k} + r_{33} + \frac{f^i t_3}{s}} \end{array} \right. \\ \left\{ \begin{array}{lcl} x_{i'+l} & = & \dfrac{kf^i[r_{11}x_{i+l} + r_{12}y_{i+l} + r_{13} + \frac{f^i t_1}{s}]}{r_{31}x_{i+l} + r_{32}y_{i+l} + r_{33} + \frac{f^i t_3}{s}} \\ y_{i'+l} & = & \dfrac{kf^i[r_{21}x_{i+l} + r_{22}y_{i+l} + r_{23} + \frac{f^i t_2}{s}]}{r_{31}x_{i+l} + r_{32}y_{i+l} + r_{33} + \frac{f^i t_3}{s}} \end{array} \right. \\ \text{with } s = r^i_{31}X + r^i_{32}Y + r^i_{33}Z + t^i_3 \text{ [6]} \end{array}} \qquad (4)$$

3.3. From Five Neighboring Views to Fluid Navigation Sequences

Let us suppose we only use five neighboring uncalibrated photographs of a static real $3D$ scene. Properly arranging these views, we can generate a complete virtual navigation sequence (figure 3) as shown for example at: *http://www.eurecom.fr/˜image/spatialisation.html*.

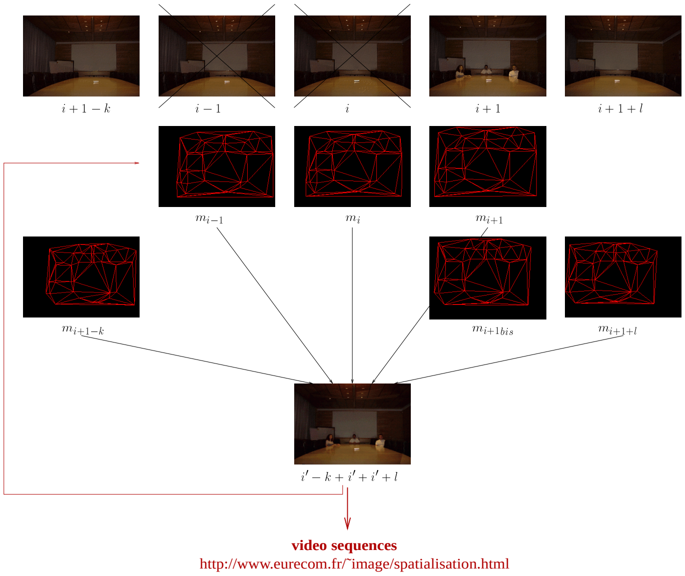


Figure 3: Generation of navigation sequences from five neighboring photographs of the real scene.

4. CONCLUDING REMARKS

To increase the frame-rate of the underlayers synthesis, we propose a new approach achieving real time, that only requires for each movement of the user the algebraic evaluation of the value of nodes of the meshes $m_{i'-k}$ and $m_{i'+l}$ (equation 4). The complete method is summarized in figure 4. In this way, we can model a real scene only by a few $2D$ uncalibrated photographs (with associated meshes for a discrete representation) and achieve real time visualization for realistic virtualized navigation in the static scene, updating the current point of view of the user at each of his/her relative movements from his/her initial position. This approach presents a real interest from the point of view of the scene acquisition (no manual CAD modeling of the scene is required), data compression (the scene model is restricted to a set of meshes and several texture views) and real time synthesis (focussing on visual realism rather than synthesis accuracy). Combining these benefits, we think the method presented here could be integrated as an interactive tool for virtualized navigation in many e_services including an immersive aspect like: e_commerce or e_conference . . .

5. REFERENCES

[1] A. Shashua. Projective Structure from Uncalibrated Images: Structure from Motion and Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(8):778–790, August 1994.

[2] O. Faugeras. *Three-Dimensional Computer Vision: A Geometric Viewpoint*. The MIT PRESS, 1993.

[3] O. Faugeras and L. Robert. What Can Two Images Tell us about a Third One? *The International Journal of Computer Vision*, 1994.

[4] K. Fintzel and J.-L. Dugelay. Visual Spatialization of a Meeting Room from $2D$ Uncalibrated Views. In *IEEE Image and Multimedia Digital Signal Processing Workshop*, Alpbach, Austria, July 1998.

[5] A. Shashua. Trilinearity in Visual Recognition by Alignment. In *European Conference on Computer Vision*, pages 479–484, Stockholm, Sweden, May 1994.

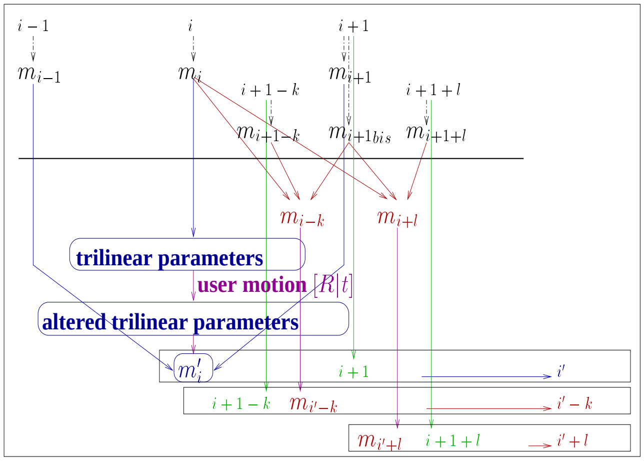


Figure 4: Final flow chart for real time natural view synthesis.

[6] K. Fintzel and J.-L. Dugelay. Manipulations Analytiques des Paramètres Trilinéaires pour la Resynthèse d'Images Inédites; restitution des Paramètres de Rotations Initiales à partir des Paramètres Trilinéaires d'un Système de Trois Caméras. Technical report, Institut Eurécom, Département Communications Multimédia, Sophia Antipolis, France, 1997. in french.

[7] R. Szeliski and H.-Y. Shum. Creating Full View Panoramic Image Mosaics and Environment Maps. In *ACM SIGGRAPH Conference*, Los Angeles, CA, August 1997.

[8] S. Peleg and J. Herman. Panoramic Mosaics by Manifold Projection. Technical report, Institute of Computer Science, The Hebrew University & David Sarnoff Research Center, Jerusalem, Israel & Princeton, USA.

[9] R. Horaud and O. Monga. *Vision par Ordinateur Outils Fondamentaux*. Hermès, 1993.

[10] K. Fintzel and J.-L. Dugelay. Virtual $3D$ Interactions between $2D$ Real Multi-Views. In *IEEE International Conference on Multimedia Computing and Systems*, Firenze, Italy, June 1999.