Trifocal tensor

Tifaine CAUMONT and Elisa PRANA

January 4, 2013

Contents

0.1 Introduction

The aim of this project was to create an application that enables to establish relations between three pictures. The first step consists in clicking on seven points on each picture representing a 3D scene from different point of view. After seven correspondances, the program is enabled to find the position of an element on the third image if you click on the position of the same element on the two others pictures.

0.2 List of the elements

Display the help in english with -h : element requested, encoded and working.

Launch into the consol with the name of the three pictures to load : element requested, encoded and working.

Launch into the consol with the name of the three lists to load : element requested, encoded and working.

Saving a list of clicked points : element requested, encoded and working.

Calculation of a tensor : element requested, encoded and working.

Transfert the points with the tensor : element requested, encoded and working.

Launch into the consol with the path of the repository in which save the lists : element not requested, encoded and working.

0.3 Description of the elements encoded

0.3.1 Tensor

A trifocal tensor is a 3x3x3 matrix. It contains 27 elements that we need to calculate. The elements T_k^{il} of the tensor have to validate the following expression :

$$\sum_{k=1}^{3} x_p^k (x_p^{'i} x_p^{''3} T_k^{3l} - x_p^{'3} x_p^{''3} T_k^{il} - x_p^{'i} x_p^{''l} T_k^{33} + x_p^{'3} x_p^{''l} T_k^{i3})$$

For each point, we have four equations. So in order to have enough equations, we need seven points. In order to find the tensor, we have to resolve the the following system : $At = 0$, where A is a 28x27 matrix and $t = (T_1^{il}, T_2^{il}, T_3^{il})$ a vector with 27 components.

Build A

The first step to resolve the system, is to build the A matrix. Let's call Ap_k^{il} the coefficient of the T_k^{il} elements in all these equations for the point p. We noticed that in each equations, only four tensor coefficients appears. The others are then equal to zero.

The rows of A corresponds to $(Ap_1^{il} Ap_2^{il} Ap_3^{il})$. For each point, there are four rows in the matrix. The first for i=1 and l=1, the second for i=1 and l=2, the third for i=2 and l=1 and the forth for i=2 and l=2.

Let's take an example with the p point. We obtain the four following equations :

$$\sum_{k=1}^{3} x_p^k (x_p^{'1} x_p^{''3} T_k^{31} - x_p^{'3} x_p^{''3} T_k^{11} - x_p^{'1} x_p^{''1} T_k^{33} + x_p^{'3} x_p^{''1} T_k^{13}) \textit{ with i=1 and l=1}$$

$$\sum_{k=1}^{3} x_1^k (x_1^{'1} x_1^{''3} T_k^{32} - x_1^{'3} x_1^{''3} T_k^{12} - x_1^{'1} x_1^{''2} T_k^{33} + x_1^{'3} x_1^{''2} T_k^{13}) \textit{ with i=1 and l=2}$$

$$\sum_{k=1}^{3} x_1^k (x_1^{'2} x_1^{''3} T_k^{31} - x_1^{'3} x_1^{''3} T_k^{21} - x_1^{'2} x_1^{''1} T_k^{33} + x_1^{'3} x_1^{''1} T_k^{23}) \textit{ with i=2 and l=1}$$

$$\sum_{k=1}^{3} x_1^k (x_1^{'2} x_1^{''3} T_k^{32} - x_1^{'3} x_1^{''3} T_k^{22} - x_1^{'2} x_1^{''2} T_k^{33} + x_1^{'3} x_1^{''2} T_k^{23}) \textit{ with i=2 and l=2}$$

So in the A matrix, we have the following values for the first equation:

$$Ap_k^{31} = x_p^k x_p^{'1} x_p^{''3}$$

$$Ap_k^{11} = -x_p^k x_p^{'3} x_p^{''3}$$

$$Ap_k^{33} = -x_p^k x_p^{'1} x_p^{''1}$$

$$Ap_k^{13} = x_p^k x_p^{'3} x_p^{''1}$$

If we adapt this to a general case, we then have :

$$Ap_k^{3l} = x_p^k x_p^{'i} x_p^{''3}$$

$$Ap_k^{il} = -x_p^k x_p^{'3} x_p^{''3}$$

$$Ap_k^{33} = -x_p^k x_p^{'i} x_p^{''l}$$

$$Ap_k^{i3} = x_p^k x_p^{'3} x_p^{''l}$$

The issue is now to fill the A matrix in the program. The rows are depending on p, i and l and the columns on k, i and l. Here is the pseudocode to fill the matrix A.

```
    Initialize A to 0
FOR p from 0 to 6
```

```
    FOR i from 0 to 1
        FOR l from 0 to 1
            FOR k from 0 to 2
                    A[4p+2i+l, 9k+3i+l]= $-x_p^k x_p'^3 x_p''^3$
                    A[4p+2i+l, 9k+3i+2]= $x_p^k x_p'^3 x_p''^l$
                    A[4p+2i+l, 9k+6+l]= $x_p^k x_p'^i x_p''^3$
                    A[4p+2i+l, 9k+8]= $-x_p^k x_p'^i x_p''^l$
                increment k of 1
              increment l of 1 END FOR
          increment i of 1 END FOR
      END FOR
increment p of 1
END FOR
```

Calculate T

Once we have the A matrix, we need to decompose it in $A = UDV^T$ with the SVD of the Eigen Library. The t vector is the last column of the V matrix. We can easily pick it up. We don't forget to put the element of t in the real tensor T with the right loops: for(int i = 0; i¡ tensor.getI(); ++i) for(int j = 0; j¡tensor.getJ(); ++j) for(int k=0; k¡tensor.getK(); ++k) tensor.setT(i,j,k, t(9*k + 3*i + j));

0.3.2 Transfert

The transfert was a difficult part of the project. First, we focus on the transfert for the first image.

Transfert on the first picture

The main difficulty was to transform the equations into matrices and vectors. At the beginning, we tried to use the SVD one the same way that for the tensor since we did'nt see that it was an equation like $Bx = b$. In order to simplify the calculations, we decided to put all the third coordinates to 1. So the matrix B was a 4x2 matrix. To fulfill the B matrix, we used the same loop than to fulfill A, removing the one with p and the one with k. Thus, in order find the eighth point of the first image, we had this :

```
FOR i from 0 to 1
    FOR l from 0 to 1
```

```
            B[2i +j, 0]= $list2(7,i)T_2^{j0} - T_i^{j0} - list2_(7,i)list3(7,j)T_2^{20} + list3(7,j)T_i^{20}$
            B[2i +j, 1]= $list2(7,i)T_2^{j1} - T_i^{j1} - list2(7,i)list3(7,j)T_2^{21} + list3(7,j)T_i^{21}$
        increment l of 1
    END FOR
increment i of 1
END FOR
```

We calculate the b vector in the same loop and apply the method solve of the Eigen Library.

Transfert on the second and third picture

We take time to understand the fact that the column of the matrix B were not depending on k anymore, but on i for the second picture and l for the third. Once understood, the method is the same unless that the k loop is added and so we have to use + = in order to sum the elements.

Once working for the three pictures, we put conditions on the click event to calculate the right transfert an change de 7 on the number of rows - 1 of the list.

0.3.3 Saving the clicked points

At first, the list were lists of 7 points and when clicking, the first point of the list was changed. It was a great idea. Then we add an eighth point, calculated with the transfert in order to keep the list of the seven points needed to build the tensor. The problem was that we did not save all the points and we were writting in the files with std::ifstream. So we changed the saveMatrix function in order to save as many points as wanted. But it still changed the initial file. We decided to save the point int tmp/list.list with a new method : " ¡¡ ". A row is just added into the matrix and the file saved for each click. Finally, we created a function (updateMatrix) in order to do the resizing of the matrix and the saving in a file.

0.3.4 Save or load in a file

Sauvegarde/Chargement dans un fichier : - Au dpart, criture dans les fichiers initiaux : bouh pas bien - Sauvegarde dans les fichiers tmp - A faire : lutilisateur peut choisir le fichier o sont enregistrs les points

Format de sauvegarde : activation du header en true dans saveMatrix()