3D Reconstruction from Two Calibrated Views

Marc Carné
Universitat Autnoma de Barcelona
Bellaterra, Catalonia/Spain
marc.carne.herrera@gmail.com

Alejandro Nespereira
Universitat Autnoma de Barcelona
Bellaterra, Catalonia/Spain
alejandronespereira@gmail.com

Gonzalo Benito
Universitat Autnoma de Barcelona
Bellaterra, Catalonia/Spain
gonzabenito@gmail.com

Abstract

A common two capture real scenes of the world is taking a picture. As we seen in previous works, a photography is a projection of the 3D world into a 2D plane. In the projection process we have some distortion in the scene captured and also valuable information is lost, as for example the depth. Despite the amount of information we loss in the projection step we are able to recover the 3D information content by a reconstruction from multiples views. In that works from two images corresponding to different point of view of scene and knowing the calibration parameters of the camera (intrinsic parameters) we can project to the 3D space points that have correspondences in both images. This method is called triangulation, and consist of find a point in the 3D space that projected on the images are the closest possible to the correspondent points that aims to reference the 3D space point. In this work that method will be presented and also we will explain how to retrieve the essential matrix that relates points in both cameras by the calibration matrix (known), decomposing the fundamental matrix that will be found by the DLT problem with the RANSAC approach. With the essential matrix the rotation matrix and the translation of the cameras can be found, what will allow us to compute depth maps.

1. Motivation

The common way to capture the 3D world by humans is to take photographies of the scenes. As it is well known, the 2D images where the real scenes are projected present a lot of limitations, the most important one is the loss of depth information and the ambiguity at the scene understanding.

Despite that limitation we are able to recover the 3D scene if we have information from different views. This means to intentionally take images from different location of the 3D world or take images from different subjects to the same 3D object. The first case is the most similar to the calibrated case because we can know the intrinsic parameters of the camera, while in the second case as each camera is different we cannot know those parameters.

In this work, based on the first case, we will be able to reconstruct a 3D scene by having different views of the same scene and also knowing the parameters of the camera (both images have been taken by the same image). The main step is to find the fundamental matrix that relates both images in pixels. With the fundamental matrix and considering the calibrated case (we know the calibration matrix, K) we can find the essential matrix, that relates points in the 2D projective space between the two cameras.

To reconstruct the 3D scene we need to know where are the camera located but the calibration matrix just contain intrinsic camera parameters. Other information as the rotation or the translation between the views can be easy retrieved from the essential matrix (E).

All this information will allow us to describe the world and be able to reconstruct 3D scenes and also estimate the depth of the points.

In this work we will see each part being able at the end to estimate depth maps.

2. Method

2.1. The Fundamental Matrix

The fundamental matrix captures the information about the epipolar geometry of 2 views. It is important to note that this matrix does not relate the pixel positions in two images of an object, as it only gives constrains about how these pixel positions change under a point of view transformation. This is not sufficient to establish a pixel to pixel relation between images, but it enormously reduces the search space for a correspondence. Moreover, as it only relates changes under view point transformations, any moving object that presents a different world position in each image can not be described with this matrix.

The fundamental matrix between two cameras is directly derived from the epipolar constraints. With two image planes and the projection of a world object into two image points **x** and **x'**, we can see the relation between the camera centers as a translation vector **T** and a rotation matrix **R**. Setting a plane that has both our camera centers and our image points **x** and **x'**, it can be deduced that the vector $T' \times Rx$ is perpendicular to this plane. As **x'** is in the plane this leads to

$$x'^T [T'_x] R x = 0 \tag{1}$$

$[T'_x]R$ is called the Essential matrix, and it relates point to point correspondences: $x'Ex = 0$.
As **x** and **x'** are both in camera coordinate system, we can obtain **p** and **p'** as:

$$p = Kx \rightarrow x = K^{-1}p \tag{2}$$

$$p' = K'x' \rightarrow x' = K'^{-1}p' \tag{3}$$

Replacing in (1) we obtain:

$$(K'^{-1}p')^T [T'_x] R K^{-1} p = 0 p' K'^{-T} [T'_x] R K^{-1} p = 0 \tag{4}$$

The fundamental matrix F is then defined as $F = K'^{-T}[T'_x]RK^{-1}$ and it relates point to point correspondences similarly to the essential matrix: $p'Fp = 0$, where **p** and **p'** are expressed in pixels. For the unlikely case where both camera intrinsic matrices are equal to the identity, the fundamental matrix is in fact, the same as the essential matrix.

2.2. 8 point Algorithm

More often than not, however, the spatial relationship between cameras or the camera intrinsic matrices are not available. However, we can estimate this with point correspondences, as we know that $p'Fp = 0$ for every pair of matching points. As both points are in homogeneous pixel coordinates we can write it as

$$\begin{pmatrix} u & v & 1 \end{pmatrix} \begin{pmatrix} F_{11} & F_{12} & F_{13} \\ F_{21} & F_{22} & F_{23} \\ F_{31} & F_{32} & F_{33} \end{pmatrix} \begin{pmatrix} u' \\ v' \\ 1 \end{pmatrix} = 0 \tag{5}$$

This can be rewritten as:

$$\begin{pmatrix} uu' & vu' & u' & uv' & vv' & v' & u & v & 1 \end{pmatrix} \begin{pmatrix} F_{11} & F_{12} & F_{13} & F_{21} & F_{22} & F_{23} & F_{31} & F_{32} & F_{33} \end{pmatrix}^T = 0 \tag{6}$$

Adding 7 additional point correspondences creates a linear system $Wf = 0$, solvable with Linear Least Square, decomposing W into its SVD matrices we can obtain the solution as the last column of V, and then use it to compose F from the f values. However, this gives a rank 3 fundamental matrix, which make the epipolar lines not coincide in the epipole. To solve this, we need to decompose again using SVD and force F to be rank 2, by removing the last singular value in the diagonal of the matrix D. With this we can recompute a F that is assured to have rank 2.

2.3. Normalized 8-point algorithm

Reliable as it is, the Linear Least Square method is very sensible to noise and highly unstable, performing poorly when faced against data of varying magnitude. This can be addressed by previously transforming the data to a normalized space. Data in this space should be centered around the origin and have a mean square distance to it of 2 pixels. This can be achieved

with an homography consisting of a translation and a scale factor. While it is simple to see that the translation is the centroid of the original data, the scale factor is more complex to obtain. The mean square distance from each point to the centroid is

$$d^2 = \frac{1}{N}\sum \|\frac{p_i}{s} - \frac{c}{s}\|^2 = \frac{1}{s^2 N}\sum \|p_i - c\|^2 \tag{7}$$

where p_i are the points in the original space and c is their centroid. The term s is the scale that we want to apply to the points. It can be proven that the scale is:

$$s = (\frac{1}{d^2 N}\sum \|p_i - c\|^2)^{\frac{1}{2}} = (\frac{1}{2N}\sum \|p_i - c\|^2)^{\frac{1}{2}} \tag{8}$$

With our centroids **c** and **c'** and the scales **s** and **s'** , we can build our pair of homographies **H** and **H'** that normalize our two input set of points:

$$H = \begin{bmatrix} s^{-1} & 0 & s^{-1}c_x \\ 0 & s^{-1} & s^{-1}c_y \\ 0 & 0 & 1 \end{bmatrix} \tag{9}$$

H' is constructed identically using **c'** and **s'**. We calculate $q = Hp$ and $q' = H'p'$, our normalized inputs and use them to obtain the fundamental matrix with the 8 point algorithm described above. The resulting F_q, however, still has to be de-normalized in order to be valid for our points **p** and **p'**:

$$F = H'^T F_q H \tag{10}$$

2.4. RANSAC Fundamental Matrix

Although normalizing the input data shields the fundamental matrix from errors due to numerical instability, errors in the points selected will result in a wrong fundamental matrix. This is due to the fact that the fundamental matrix estimation is naive in the sense that it does not perform any validation of the selected matches. To solve this, an iterative RANSAC approach can be used to estimate a matrix that provides a good result while being robust to outliers.

The idea is very similar to the one presented in the past session for homography estimation. From our set of point matches we select a subset of N points (8 for fundamental matrix estimation), from which we compute **F**. We test this matrix against all our points and count the number of inliers. We repeat this until we are sure, in a probabilistic way, that we have picked at least one subset free of outliers. As with the homography case, we need to provide a method to compute the number of inliers for a given model. We have applied a threshold over the first order of the geometric error to determine if a point correspondence is an inlier. This approximation is also known as the Sampson distance:

$$d_i = \frac{(x_i^T F x_i)^2}{(Fx_i)_1^2 + (Fx_i)_2^2 + (Fx_i')_1^2 + (Fx_i')_2^2} \tag{11}$$

where the subindex denotes the dimension.

2.5. Epipolar geometry

Through the session we have used several notions of epipolar geometry that we think is necessary to add here.

2.5.1 Epipoles

The epipoles present very useful relationships with the rest of the epipolar geometrical elements. The epipoles are the points where the baseline intersect with the image planes, and therefore $Fe = 0$ and $F^t e' = 0$. Moreover, the epipoles are the points in which all the epipolar lines intersect, and thus it can be written that $l \times e = 0$ for all epipolar lines l.

2.5.2 Relationship between a point and its epipolar line

From the fundamental matrix defined as $x'Fx = 0$, we can define the relationship between a point in the first image **x** and its corresponding epipolar line in the second image as: $l' = Fx$. The relationship between a point in the second image **x'** and its epipolar line in the first image is $l' = F^T x$.

2.6. Essential matrix

This matrix is similar to the fundamental matrix, but in this case relates points in the projective space between images instead of directly pixels. As both matrices are related, we can found the essential matrix from the fundamental matrix, knowing the calibration camera matrices.

$$K'^{-1}EK^{-1} = F \tag{12}$$

From the last relation we can aisle the essential matrix allowing us to compute it from the calibration matrices and the fundamental matrix.

2.7. Rotation matrix

Once the essential matrix, that relates points in the projective space between the two cameras we can extract the rotation matrix and the translation vector. The main step to compute both matrices is to decompose the essential matrix in two orthogonal matrices and a diagonal matrix. This can be computes as the singular value decomposition of the essential matrix. The singular matrix decomposition of the essential matrix is expressed as:

$$SVD(E) = UDV^T \tag{13}$$

A main characteristic of the essential matrix is that has rank equal to two, so it has two eigenvalues different from zero and with the same value. For the singular value decomposition we can find the eigenvalues in the non-zero coefficients of the diagonal matrix D. This matrix is up-to-scale.

From the decomposition we can define the rotation matrix as:

$$R = UWV^T or R = UW^TV^T \tag{14}$$

Where matrices U and V comes from the decomposition and W is a skew symmetric matrix that can be written as:

$$W = \begin{bmatrix} 0 & -1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{15}$$

By this way we have two possible rotation matrices that are equal to a 180 degrees of rotation with respect to the baseline.

2.8. Translation vector

That vector corresponds to the translation of position between both cameras. To find that vector we can extract it directly from the decomposition. In this case we will take the last column of the matrix U that comes from the singular value decomposition.

2.9. Triangulation

With all the previous procedure we are able to find the projection matrices for the both cameras. In this case to simplify the camera representation we will consider one camera as a reference and we will compute the movement of th camera with respect to the reference one

Triangulation method consist in 3D points in the space from the 2D points of the images (that can be computed with matching techniques as SIFT) and the projection matrices.

The main idea of the method consist in find a point in the 3D space that minimize the distance between the projection of this 3D point on the images and the true position of the point.

To compute the world points from the image correspondences, we will build a matrix A from the image points and the projection matrix:

$$A = \begin{bmatrix} xp^{3T} - p^{1T} \\ yp^{3T} - p^{2T} \\ x'p'^{3T} - p'^{1T} \\ y'p'^{3T} - p'^{2T} \end{bmatrix} \tag{16}$$

In the last expression 'x' and 'y' are the coordinates of the points in both images that comes from the corresponding points, and the projection matrices P and P' have been expressed as rows.

$$P = \begin{bmatrix} P^{1T} \\ P^{2T} \\ P^{3T} \end{bmatrix} \tag{17}$$

This problem can be solve as a matrix decomposition doing the singular value decomposition as the problem can be defined as:

$$AX = 0 \tag{18}$$

2.10. Re-projection error

Once performed the triangulation we have found the 3D points that corresponds to the image matches, we can measure the error between the projection of the 3D space to the image and the true point (considering the true point as the point we used to compute the 3D point).

This error can be computed as the distance between both points, the re-projected one (from the 3D point) and the true point, with the geometric distance or the Sampson error.

3. Results

3.1. Test the 'triangulate' function

In the first exercise we have to check that the function *trangulate* to be written performs as desired. This function is based on the homogeneous linear method approach in which we write the problem as seen in eq. 18, Where A is defined by eq. 16. For this operation we need at least 8 points to cope with the degrees of freedom of the problem. The test 3D points are randomly generated, then projected with each *P* matrix, and finally triangulated. The result is contrasted with the original test batch, so as to confirm that the triangulation error is low.

3.2. Reconstruction from two views

This is the main section, in which two vies will be used to reconstruct the 3D points. The first step consists in reading the images, extracting key points and computing the matches between the two views' keypoints. Figure 1 shows the matching of the SIFT keypoints.

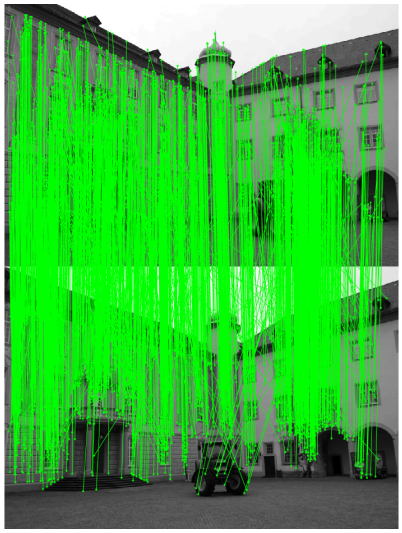

Figure 1: The sift keypoints found at the two images match-up.

Following the SIFT matching, comes the employment of RANSAC in order to obtain the fundamental matrix from the matching points. Then again, we search for a transformation matrix that allows us to fit the most of the sift correspondences found.

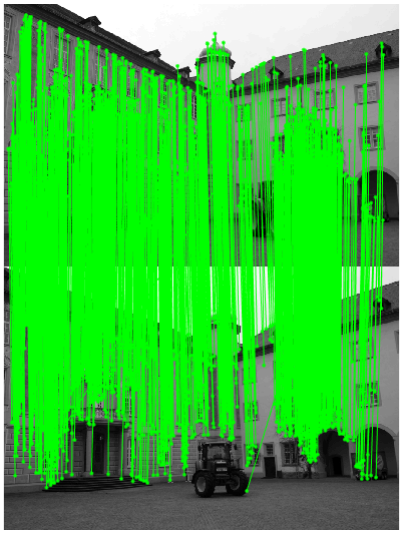

Figure 2: The matches for our SIFT keypoints. To ease the viewing, each match line has been coded depending on angle between both image pixel positions. It can be seen that most of them share a very similar angle, which points to a robust matching. The few exceptions are caused by the keypoints not being present in both images.

We then obtain the fundamental matrix **F** which will let us employ the geometric methods:

$$F = \begin{bmatrix} -2,39827e-08 & 2,35269e^{-}06 & -0,00119 \\ -1,52798e-06 & -1,44769e-07 & -0,0035 \\ 0,00074 & 0,002981 & 0,29028 \end{bmatrix} \tag{19}$$

At this point, we are ready to compute the second camera matrices candidates. We are given the camera calibration matrix *K*. With it and the fundamental matrix, we can then obtain the *essential* matrix $E = K' * F * K$. E is used to get a SVD decomposition, and with U and V, calculate the possible second camera's projection matrices. We will obtain 4 candidates that can be seen in Fig. 3.

Now, we have obtained 4 possible cameras to be the providers of the second view, but only one is correct. To figure this out, we make use of the fact that given a certain point **X** in the space, it will have a projection $x = P1 * X$ in the image 1 and given that our coordinate system's origin has been placed in camera 1, x's z value will be positive -meaning that the 3D point is located in front of the camera-. With this information we will find one of the camera candidates that also makes the X point to fall in front of it -meaning that the projection of the 3D point in the view of the camera will hold a z value with the same sign as the camera 1's-.

Once we separate our camera 2 with our projection matrix 2, we compute the 3D projection of the point matches between the two views. The result can be seen in Fig. 4 and Fig. 5.

3.3. Depth map computation with local methods

In this task we make use of local disparity ponderation methods in order to define a depth map. First of all, we test our own stereo-computation function based on Sum of Squared Differences. This function basically computes the disparity between two rectified views according to a cost function. The initial implementation uses the Sum of Squared Differences as cost function, computing this value on a window over each image, horizontally rotating the right image's window in order to find the disparity value that lowers the most the cost for that pixel. Figures 6, 7 and 8 display the results over a dummy pair of images when selecting various window sizes. When using small window sizes, the depth mapping takes into account very subtle changes in intensity as a complete different depth. On the contrary, a big window size, may generate the loss of some depth details -such is the case of the arm of the lamp which tends to disappear-. The bigger the area, the more difficult for a

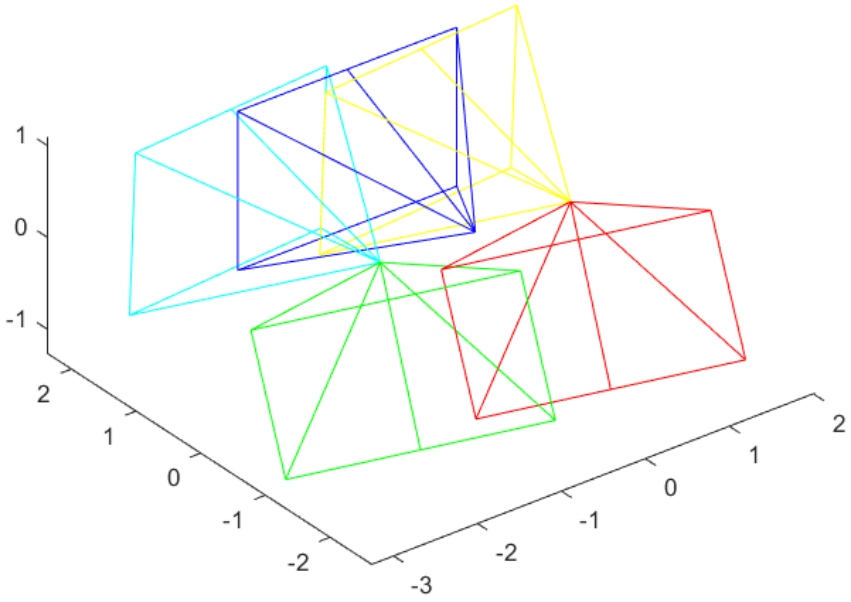

Figure 3: Plot of the camera 1 (blue) and the four candidates to camera 2.

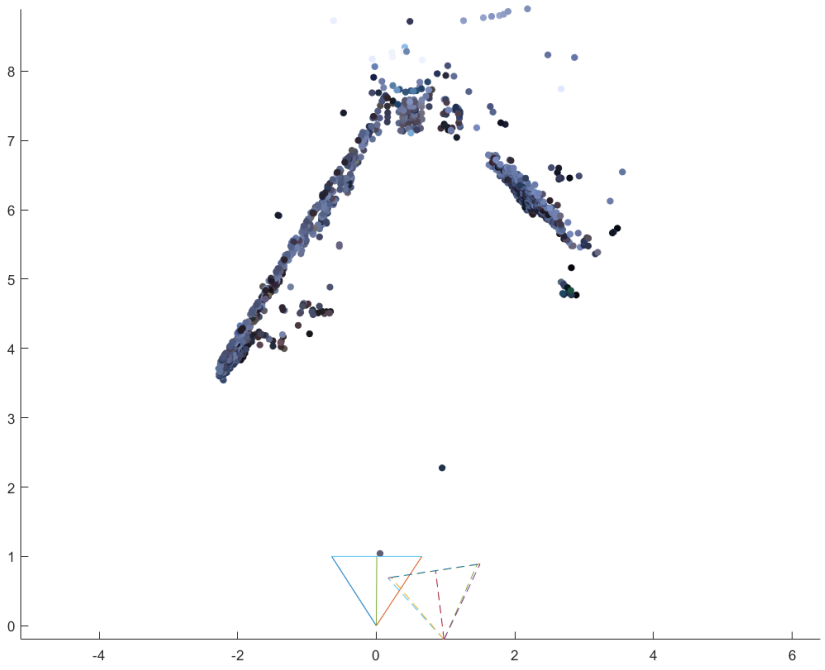

Figure 4: Structure reconstruction from two views, seen from atop. Notice the second camera with stripped line.

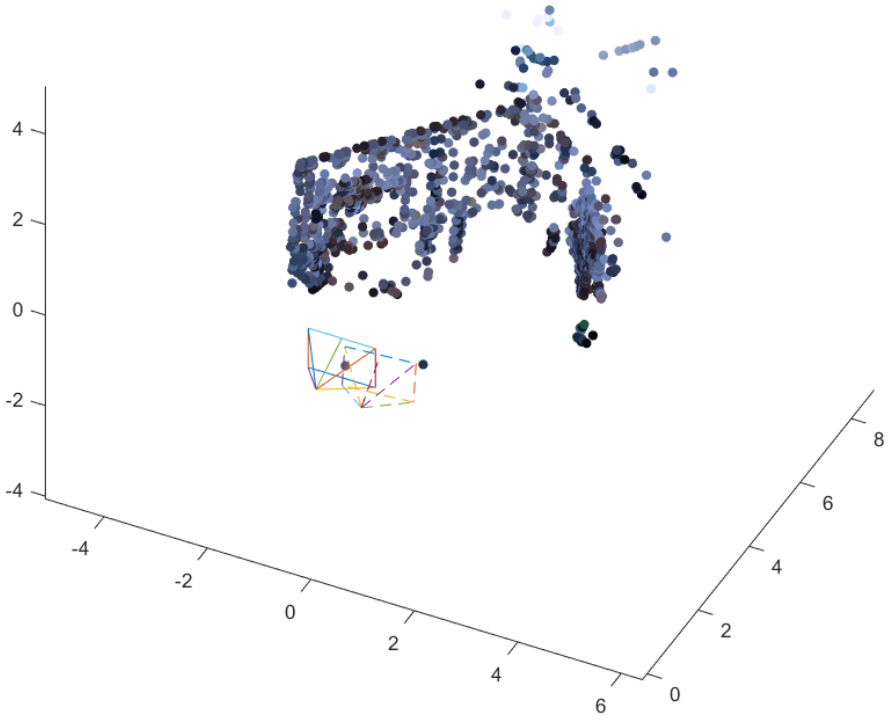

Figure 5: Structure reconstruction from two views, 3d visualization. Notice the second camera with stripped line.

relative small change in intensity to be traduced as a depth value, as the window acts like a smoothing filter for the disparity is dominated by the most recurrent values.

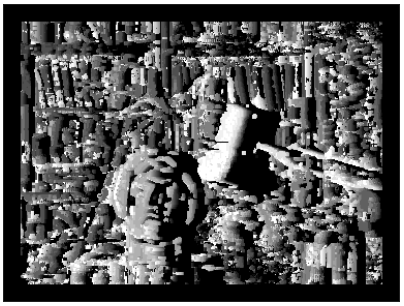

Figure 6: Depth mapping using disparity through SSD cost and window size 3x3.

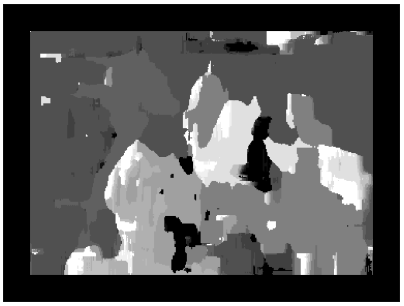

Figure 7: Depth mapping using disparity through SSD cost and window size 21x21.

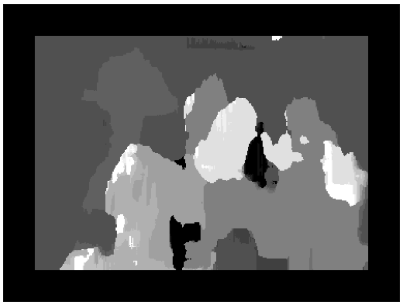

Figure 8: Depth mapping using disparity through SSD cost and window size 31x31.

Furthermore, we experimented with a different cost function to see what effects it had in the outcome. Figures 9, 10 and 11 show the result of NCC cost implementation. Again we can see that the smaller the window, the the finner the the structures that can be noticed.

These results let us see that NCC is a method which allows a little better depth homogeneity thanks to the fact of the mean value extraction within the window. This allows to compare normalized windows and penalize further the occurrences of high depth disparity. Nevertheless, this same 'smoothing' affects a good contours definition.

Finally, we experiment with the facade rectified images. The results shown in Fig. 12, 13 and 14 reinforce what has been seen with the samples: small windows are very susceptible to subtle intensity changes, making the output very granulated

Figure 9: Depth mapping using disparity through NCC cost and window size 3x3.

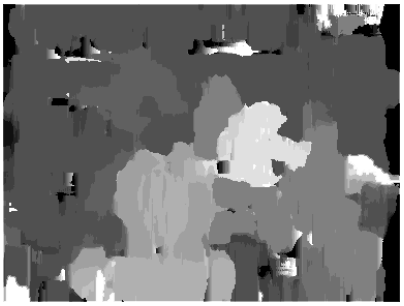

Figure 10: Depth mapping using disparity through NCC cost and window size 21x21.

Figure 11: Depth mapping using disparity through NCC cost and window size 31x31.

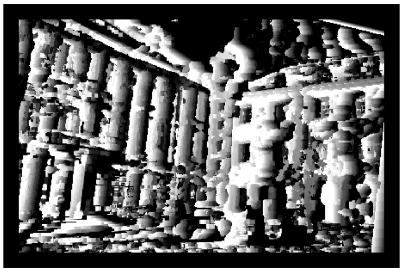

Figure 12: Depth mapping using disparity through SSD cost and window size 3x3.

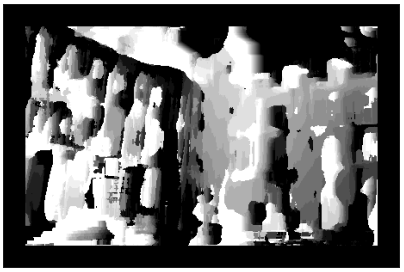

Figure 13: Depth mapping using disparity through SSD cost and window size 21x21.

Figure 14: Depth mapping using disparity through SSD cost and window size 41x41.

and not with clear depth segmentation.

4. Conclusions

To sum it all, we have performed a series of steps that allow structural reconstruction of 3D scenes from 2D views. We applied concepts which help to define the importance of aspects such as intrinsic camera parameters and extrinsic parameters as well as each one's influence in projective geometry and stereo vision. Beginning with the triangulation of the points in space, following with the construction of the fundamental matrix, the definition of the candidates for camera 2, to the reconstruction of the 3D structure based on 2D views, we could grasp the basis of spatial positioning from stereo vision, and also address in a naive way the task of depth perception from stereo images.

5. Available code

All the code implemented in this project can be found in the following github repository: https://github.com/marccarne/M6project