Inkless Painting

Emanuele Ghelfi
Politecnico di Milano
emanuele.ghelfi@mail.polimi.it

Emiliano Gagliardi
Politecnico di Milano
emiliano.gagliardi@mail.polimi.it

September 17, 2018

1 Introduction

Wouldn't be great to write or draw on any surface, without the need of ink, and obtain the result in digitalized format? Well, we think it would. This is why the "Inkless Painting" project was our choice among the ones proposed by our Image analysis and Computer Vision professor.
The task was to develop an algorithm that, given a video of someone drawing using a pen without ink, recovers the 3D trajectory of the pencil tip. Given that, it is possible to reconstruct the drawing, by simply keeping the part of the trajectory near the writing surface.
We obtained good results by imposing some requirements:

- The user should provide calibration images together with the video: at least twenty or thirty for intrinsic calibration, while a single one for extrinsic calibration. Calibration images are pictures of a known template, that will be called *calibration marker* in this document.
- The camera pose need to be fixed during the whole video, and equal to the one in which the extrinsic parameters calibration image was taken.
- The pencil must be rigidly attached to a simple marker we designed, that we call *pencil marker*, and visible during the whole video. Best results are obtained when the marker is attached to a planar surface on the pencil.
- A color space filter is given. This should preserve, on each frame, the marker and few other aspects of the image. From this, a robust detection algorithm will extract the marker geometrical features on each frame.
- A model of the pencil is given. The model consists in the length of some lines belonging to the pencil marker, and the tip position with respect to some point of the marker.
- The pencil is always directed toward the table.

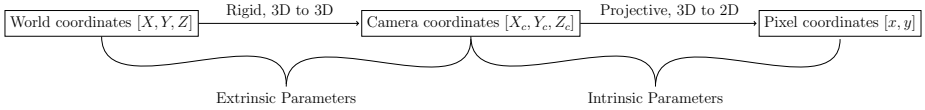
A very important aspect is that we do not use any tracking algorithm: the marker is detected and the 3D position of the pencil tip is recovered frame by frame, without using any prior information over the pencil pose in the previous frames. At the end, when a full trajectory is recovered, we apply a very simple smoothing procedure with outliers rejection.
Summarizing, the aim of our work is to show the application of Computer Vision technologies to a particular task of everyday life. With some enhancement we describe in section 8, we think that it is possible to obtain a user friendly application based on our reconstruction algorithm.
The document is structured in the following way: in section 2 we describe the calibration algorithm used, in section 4 how the salient points of the pencil marker are extracted from a frame, and in section 5 how the 3D reconstruction of those points is performed. Then we describe the smoothing and outliers rejection procedure in section 6. Finally in section 7 we show the result of the application of our pipeline to some videos.

Nomenclature

$\mathbf{K}$	Calibration matrix
$\mathbf{o}$	position of the camera reference frame in the world reference frame, as column vector
$\mathbf{P}$	projection matrix
$\mathbf{R}_{\mathbf{c}}^{\mathbf{w}}$	rotation of the camera reference frame with respect to the world reference frame
$\mathbf{R}_{\mathbf{w}}^{\mathbf{c}}$	rotation of the world reference frame with respect to camera reference frame
$\mathbf{t}$	position of the world reference frame in the camera reference frame
$\mathbf{x}$	image point in pixel coordinates, as column vector
$\mathbf{X}^f$	3D point coordinates expressed in reference frame f, as column vector
c_x, c_y	Optical center
f_x, f_y	Focal length
s	Skew factor

2 Calibration

The calibration phase aims at finding intrinsic and extrinsic camera parameter, i.e. the calibration matrix $\mathbf{K}$, the rotation matrix $\mathbf{R}_{\mathbf{w}}^{\mathbf{c}}$ and the vector $\mathbf{t}$ (Hartley and Zisserman, 2003). Calibration is needed to understand how points in the real word are projected in the image plane, to measure the size of an object in world units, or to determine the location of the camera in the scene.



It is worth noticing that our calibration and localization procedure is free from user interaction. The user should only provide calibration images and one localization image. Our algorithm establishes when a calibration image is good and it is able to detect bad images, discarding them.

Intrinsic Parameters The intrinsic parameters include the focal length $[f_x, f_y]^t$, the optical center $[c_x, c_y]^t$, also known as the principal point, and the skew coefficient s. The camera intrinsic matrix, $\mathbf{K}$, is defined as:

$$\mathbf{K} = \begin{bmatrix} f_x & s & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} . \tag{1}$$

The intrinsic parameters represent a projective transformation from the 3D camera's coordinates into the 2D image coordinates

Extrinsic Parameters The extrinsic parameters consist of a rotation, $\mathbf{R}$, and a translation, $\mathbf{t}$. The origin of the camera's coordinate system is at its optical center and its x and y-axis define the image plane. The extrinsic parameters represent a rigid transformation from 3D world coordinate system to the 3D camera's coordinate system.

2.1 Calibration From Homographies

Among all the possible methods for camera calibration present in the literature (Caprile and Torre, 1990; Brown, 1971; Hartley, 1994; Maybank and Faugeras, 1992), we follow the approach of calibration from homographies (Zhang, 1998). Given an image of a known planar scene we can place the world reference frame on the plane π describing the scene, thus $\mathbf{R}_\pi = \mathbf{R}_w^c$, for ease of notation we denote it by $\mathbf{R} = [\mathbf{r}_1\ \mathbf{r}_2\ \mathbf{r}_3]$. A point on the plane π is described in the world

reference frame with $[X\ Y\ 0\ 1]^t$ and it is related with its image $[u\ v\ 1]^t$ by:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = s\mathbf{K}\left[\mathbf{R} \mid \mathbf{t}\right] \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} \tag{2}$$

$$= \mathbf{K}\begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{3}$$

$$= \mathbf{H}\begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} . \tag{4}$$

Thus the knowledge of the homography $\mathbf{H}$ is needed for the estimation of $\mathbf{K}$. From the calibration images we extract SURF (Bay et al., 2008) features, we match extracted features with features of the known template and finally we estimate the homography using RANSAC (Fischler and Bolles, 1981), for robust model fitting.
Circular points on the plane π have coordinates $I = [1\ +j\ 0]^t$ and $J = [1\ -j\ 0]^t$. Their image belongs to the image of the absolute conic $\omega = \left(\mathbf{K}\mathbf{K}^t\right)^{-1}$. Let $\mathbf{H} = [\mathbf{h}_1\ \mathbf{h}_2\ \mathbf{h}_3]$, we can write:

$$\mathbf{h}_1^t\omega\mathbf{h}_2 = 0 \tag{5}$$

$$\mathbf{h}_1^t\omega\mathbf{h}_1 = \mathbf{h}_2^t\omega\mathbf{h}_2\,. \tag{6}$$

We fit an homography for each image, thus we need at least two images for solving the four degrees of freedom of $\mathbf{K}$, assuming no skew. For a better fitting we used at least twenty images and least square solution. In our implementation we directly parametrized ω, noting that, for a zero-skew camera ($s = 0$), this can be rewritten as:

$$\omega = \begin{bmatrix} \alpha^2 & 0 & -c_x\alpha^2 \\ 0 & 1 & -c_y \\ -c_x\alpha^2 & -c_y & f_y^2 + \alpha^2 c_x^2 + c_y^2 \end{bmatrix}, \tag{7}$$

where $\alpha = \frac{f_x}{f_y}$ is the aspect ratio of the pixels.

(a) The calibration marker

(b) An example of calibration image

Figure 1

2.2 Extrinsic Calibration

Camera localization is the process of determining extrinsic camera parameters. Our localization process requires the image of the calibration marker to be taken from the same position as the one during the video.
Placing the world reference frame on the calibration marker, we apply the planar object localization algorithm (see appendix A) obtaining the rotation $\mathbf{R}_w^c$ and translation vector $\mathbf{t}$. The required homography is estimated as in the previous section.
The camera location is determined by the inverse transformation:

$$\mathbf{R}_c^w = \mathbf{R}_w^{c\,-1} \tag{8}$$

$$\mathbf{o} = -\mathbf{R}_c^w \mathbf{t}\,. \tag{9}$$

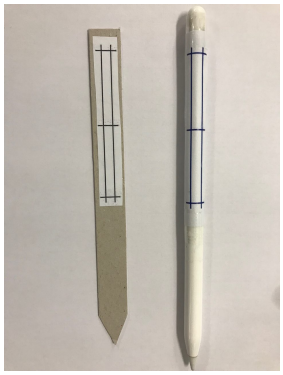
3 The Pencil Marker

Pencils present a very limited surface, thus a custom marker needs ot be designed for our purposes. Plus, some requirements need to be satisfied:

- The marker needs to be composed by few well distinguishable basic elements, for easy solution of the data association problem.
- The marker can be easily detected and should present keypoints that can be fed to a single view 3D reconstruction algorithm.
- Given the marker keypoints 3D reconstruction, the pencil tip position is uniquely determined.

We came out with a planar marker composed by five lines of the same color, that can be isolated from the rest of the image in some color space.
Among the five lines two are long and parallel, while the other three are shorter and perpendicular to the long ones. The long lines intersect the short ones, generating a triple of collinear points for each vertical line. This choice is motivated by the fact that a triple of collinear points, whose real world distances are known, can be 3D reconstructed by the algorithm presented in appendix B. We position the marker on the pencil in such a way that the long lines are parallel to the pencil axis, and the tip projection on the marker plane falls between the long lines.
The presence of two triple of collinear points placed on parallel lines allows to robustly fit the 3D vertical direction of the marker, and gives a better understanding of the pencil position. In particular, a single triple of vertical points would not allow to recover the tip position, giving infinite solutions for the rotation around the three points axis.

(a)



(b)

Figure 2: Our pencil marker (a), and two examples of pencils with a marker placed on them (b).

4 Detection

The goal of this phase is to extract, from each frame, the two triple of collinear points belonging to the long lines of the *pencil marker*. Inside each triple data association is solved, distinguishing points as *high*, *med*, and *low* point.
In our approach detection is divided into: preprocessing, line extraction with Hough transform, line selection, Harris Corner Detection and finally marker fitting.

4.1 Preprocessing

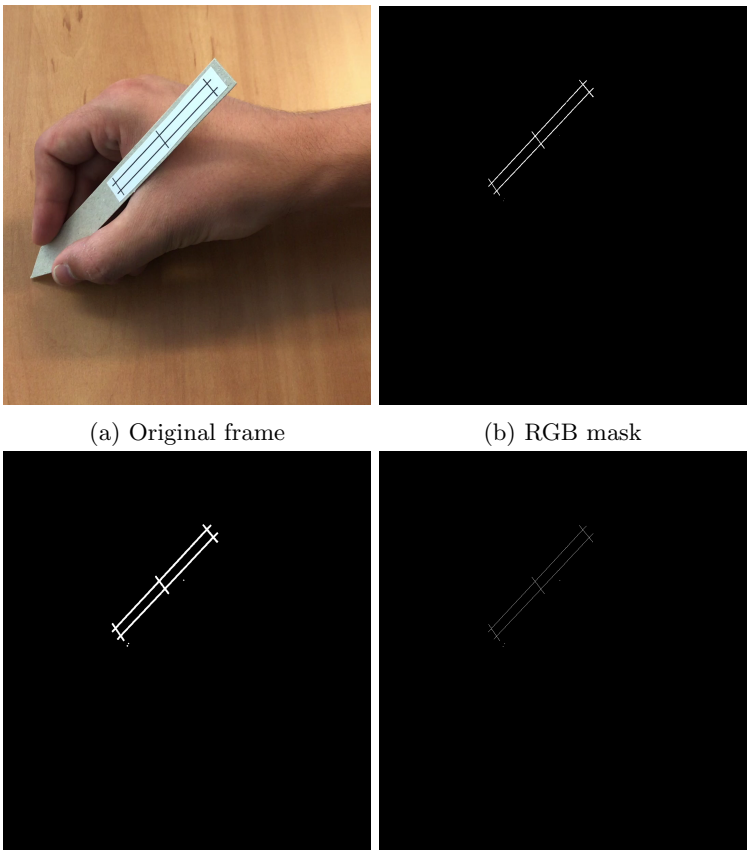
During the preprocessing phase a color space filter is applied to the frame removing all useless parts of the image and saving (hopefully) only the marker and few other features.
We further apply a dilation filter followed by an erosion filter with infinite width (see appendix D). In this way we obtain lines of approximately one pixel width, helping the Hough algorithm in its work.
An example of the preprocessing pipeline intermediate results is shown in fig. 3.

4.2 Hough Transform

Horizontal and vertical lines on the pencil marker are independently extracted, so that two different set of parameters (line length, angles, neighborhood suppression) can be set. Hough parameters must be tuned on the specific video, but it is worth noting that we do not expect only the interesting lines to be extracted, since we apply a robust selection algorithm.
This module outputs two sets of marker lines candidates, one for long vertical lines, and the other for short horizontal lines.

4.3 Line Selection

This module selects, among the candidates, the lines most likely to be the ones of the marker.
First we consider all the possible couples of vertical lines, and we remove the ones whose elements intersect, when considered as segments. This can be done since Hough lines are segments in the image.
Among the remaining, we select the couple with smallest angular distance. This is based on the assumption that the perspective effect is not so evident for couple of short near lines, at small distance from the camera.
Given the pair of vertical lines, we search for the triple of horizontal lines nearest to be perpendicular to the vertical couple, with the same constraints on the intersection point as before.
Results are showed in fig. 4. It is possible to see that there are many horizontal candidates after having applied Hough. Our algorithm is able to select the lines that most likely belong to the marker getting rid of useless lines.



(a) Original frame (b) RGB mask

(c) Dilated Image (d) Eroded Image

Figure 3: An example of intermediate results of the preprocessing pipeline

4.4 Harris Corner Detection

Lines on the pencil marker are fully determined by the six points at the intersection of the five lines. Since our reconstruction algorithm is based on these points, it is important to detect them as precisely as possible.
At a first glance we could use intersection of lines detected in the previous stage, but this can be inaccurate since lines originates from the application of a color space filter, that suffers lighting variation. For a more precise keypoint detection we use the lines intersections as guesses, and we search corners in their neighborhood.
Given $\mathbf{x}^i_{hint}$ the i-th intersection, we apply *Harris Corner Detection* to a small region of the image centered in $\mathbf{x}^i_{hint}$. We extract the n strongest corner in that region and average them 5. This adjustment gives a big improvement with respect the use of the lines intersections.

4.5 Marker Fitting

All the previous phases aim at finding good corner candidates, but they do not take in account the exact marker model. Thus it is possible to obtain non-collinear points, a situation not admissible for the following stages.
Given the marker model expressing the 2D relative positions of marker points, it is possible to find an homography $\mathbf{H}$ mapping them to the ones detected in the current frame. $\mathbf{H}$ can then be applied to the marker model, obtaining the least square marker image given the detected corners. This stage also acts as 2D geometrical check: if the homography found is too poor (the residual error is too high) the detection phase fails and the frame is rejected.
All the detection pipeline is specified in algorithm 1.

Algorithm 1 Detection

Input: frame f, pencil marker model $\mathbf{m}$, 2D error threshold ϵ, $applyColorFilter$ function
Output: Image of marker corner points $\mathbf{x}_c^i,\ i = 1..6$ if geometrical check passed, otherwise frame refused

$mask \leftarrow applyColorFilter(f)$
$mask' \leftarrow dilateAndErode(mask)$ ▷ Preprocessing
$lines \leftarrow HoughLines(mask')$
$verticalLines, horizontalLines \leftarrow selectLines(lines)$
for i in 1,..,6 **do**
    $\mathbf{x}_{int}^i \leftarrow$ i-th intersection between horizontal and vertical lines
    $\mathbf{x}_h^i \leftarrow HarrisCornerDetector(ROI(\mathbf{x}_{hint}^i))$
end for
$\mathbf{H} \leftarrow fitHomography(\mathbf{m}, \mathbf{x}_{int})$ ▷ Marker Fitting
for i in 1,..,6 **do**
    $error_i = || \mathbf{x}_{int}^i - \mathbf{H}\mathbf{m}^i ||$ ▷ Residual
    if $error_i > \epsilon$ **then** ▷ 2D geometrical check
        refuse frame
    else
        $\mathbf{x}_c^i \leftarrow \mathbf{H}\mathbf{m}^i$ ▷ Corner replacement with the fitted one
    end if
end for
return Detected Corners in frame f: $\mathbf{x}_c$

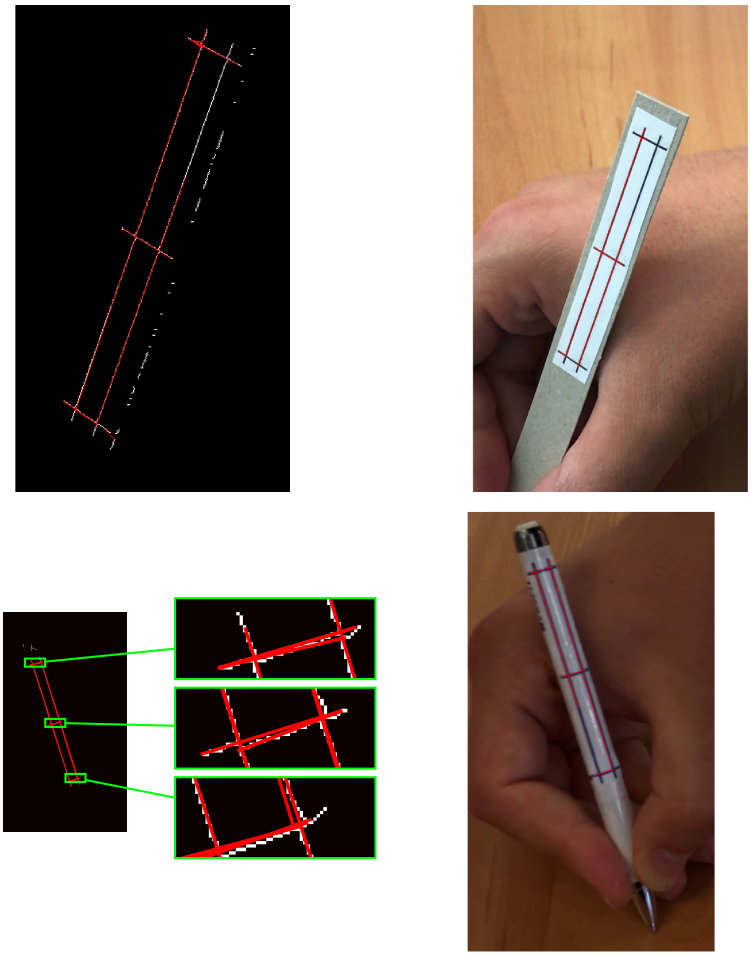


Figure 4: Two examples of line extraction. Left: lines extracted by Hough. Right: lines selected by our algorithm

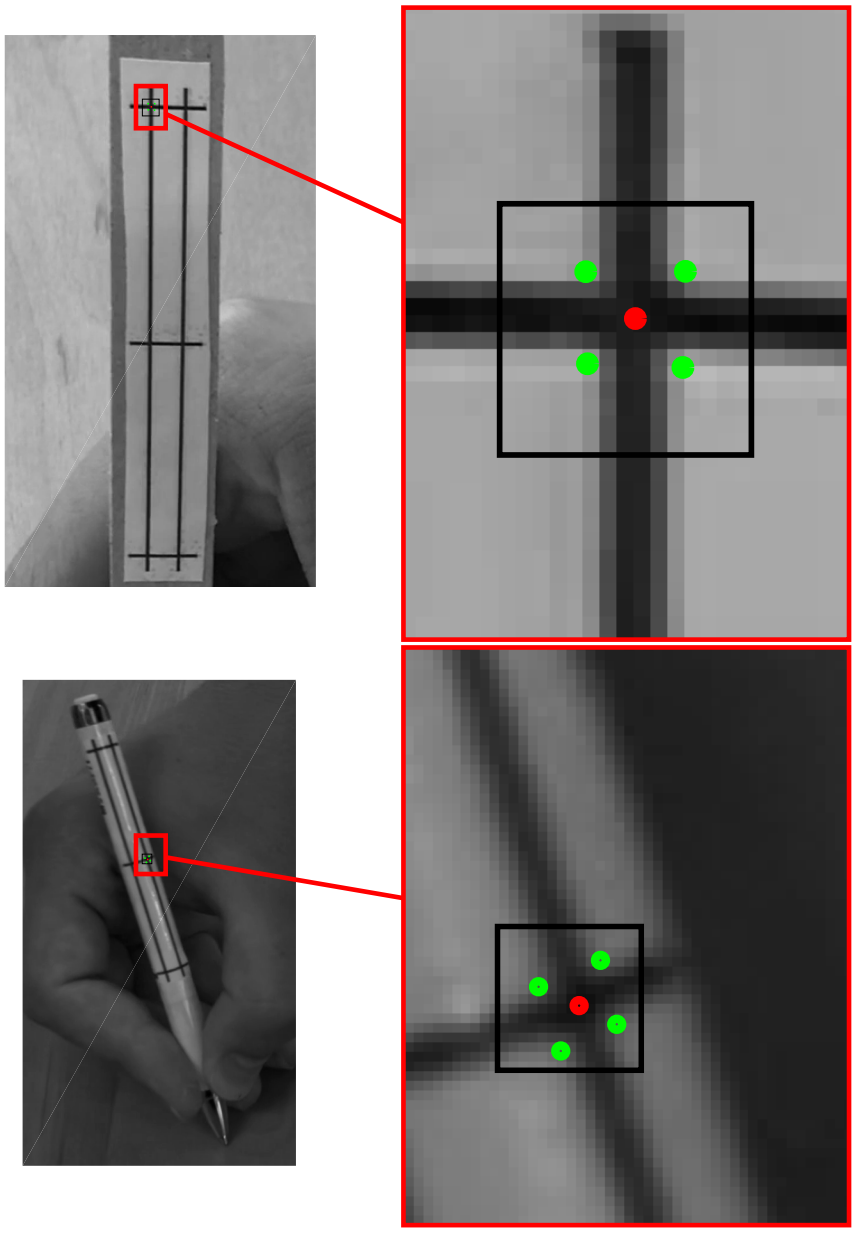


Figure 5: Two examples of corner detection. A zoom of the marker in grayscale (left). Region of interest (right) with extracted corners using Harris corner detector in green, the average of the coordinates of the corners in red. The black square represents the region of interest centred in the line intersection.

5 Reconstruction

We want to recover, from the knowledge of the marker points pixel coordinates $\{\mathbf{x}_i^{up}, \mathbf{x}_i^{med}, \mathbf{x}_i^{low}\}_{i=1,2}$, the tip 3D position $\mathbf{X}_t^c$.
Since the marker positioning on the pencil is known, the tip coordinates can be easily computed from the marker points $\{\mathbf{X}_i^{c,up}, \mathbf{X}_i^{c,med}, \mathbf{X}_i^{c,low}\}_{i=1,2}$. Thus the most substantial step of this module is the marker reconstruction. We consider two different methods for reconstruction: marker reconstruction from homography (section 5.1) and marker reconstruction with cross-ratio (section 5.2).

5.1 Marker Reconstruction with Homograhy

This method is a trivial application of the planar object localization algorithm (see appendix A). Since the marker is planar, we can model it with six 2D points, and fit the homography $\mathbf{H}$ that brings them to their known image in the current frame. The model can be thought, for instance, as a set of six points $\{\mathbf{X}_i^{o,up}$, $\mathbf{X}_i^{o,med}, \mathbf{X}_i^{o,low}\}_{i=1,2}$ whose third coordinate is 0 in some reference system o.
From $\mathbf{H}$ we obtain the rototraslation $\mathbf{T}$ that aligns o axis to c axis. The expression of the marker points in the camera frame is obtained applying the transformation $\mathbf{T}$ to their corresponding expressions in o.

5.2 Marker Reconstruction with Cross-ratio

This method is based on the single view 3D reconstruction algorithm explained in appendix B, that reconstructs a triple of collinear points whose real world distances are known.
It is now useful to briefly recap it, for a simpler explanation of how we adapted it to our settings.
First the vanishing point image $\mathbf{v}'$ is found, imposing cross ratio conservation and collinearity ($\mathbf{v}'$ belongs to the line through the three points). The 3D direction of the triple is the back-projection ray of $\mathbf{v}'$.
The back-projection rays of the three points form, with the 3D line through their corresponding 3D elements, a system of triangles whose angles and some sides length are known. From there the points can be reconstructed using triangles geometry.

Vanishing point computation Since our marker presents two triples on parallel lines, we can improve the vanishing point estimation using them jointly: we end up with an over-constrained non-linear system of four equations in two unknowns that we solve using the Levenberg-Marquadt method (Marquardt, 1963), (Levenberg, 1944).

Reconstruction As the marker orientation is known, an easy approach would be to obtain the reconstruction of a single point for each triple via triangles geometry, and obtain the remaining points moving along the marker direction of

the known lengths [1].
This method has shown in our experiments to be highly prone to noise. The simple explanation is that, for each triple, only two of the three back-projection ray determines the solution and not all the available information is exploited.
We improve the robustness by independently reconstructing each point with triangle geometry. This involves all the back-projection rays, and different combinations of them in angles computation.

Fitting Up to now the reconstruction algorithm does not guarantee that the result is a real marker. We correct this by finding the least square solution.
Let $\{\mathbf{X}_i^{o,up}, \mathbf{X}_i^{o,med}, \mathbf{X}_i^{o,low}\}_{i=1,2}$ be the marker model expressed in some reference frame o, and $\{\hat{\mathbf{X}}_i^{c,up}, \hat{\mathbf{X}}_i^{c,med}, \hat{\mathbf{X}}_i^{c,low}\}_{i=1,2}$ the estimated marker points. We apply absolute orientation (see appendix C) to obtain the transformation $\mathbf{T}$ that best aligns the first set of points to the second one. $\mathbf{T}$ is applied to the model, obtaining the least square marker $\{\mathbf{X}_i^{c,up}, \mathbf{X}_i^{c,med}, \mathbf{X}_i^{c,low}\}_{i=1,2}$.
For a further geometrical check we compute the values

$$\left\| \mathbf{X}_i^{c,p} - \hat{\mathbf{X}}_i^{c,p} \right\| \quad \text{for} \quad i = 1,2 \quad p \in \{up, med, low\} \tag{10}$$

and we discard the current frame if one of these values is higher than a fixed threshold.

5.3 From Marker Points to Pencil Tip

Using one of the previous methods we can reconstruct marker points. Now we need to recover the 3D position of the pencil tip.
Note that the marker presents two symmetries: a vertical one and an horizontal one. These reflect in ambiguity in data association. The horizontal symmetry is solved in the detection step, while there is still ambiguity in the horizontal direction. This means that we do not know, for instance, if the $\mathbf{x}_1^{low}$ point belongs to the left or right triple of marker points.
Indeed, the two different data associations correspond to two possible position of the pencil, in which the one with smaller $\mathbf{X}_t^c$ third coordinate (depth in camera frame) coincides with a marker occlusion. Thus we simply solve the ambiguity due to vertical symmetry of the marker by excluding it.
Given that, we define:

$$\mathbf{X}^{c,low} = \frac{\mathbf{X}_1^{c,low} + \mathbf{X}_2^{c,low}}{2} \tag{11}$$

$$\mathbf{X}^{c,med} = \frac{\mathbf{X}_1^{c,med} + \mathbf{X}_2^{c,med}}{2} \tag{12}$$

And obtain the projection $\mathbf{X}_t^{c'}$ of the pencil tip on the marker plane as:

$$\mathbf{X}_t^{c'} = \mathbf{X}^{c,low} + \frac{\mathbf{X}^{c,low} - \mathbf{X}^{c,med}}{\|\mathbf{X}^{c,low} - \mathbf{X}^{c,med}\|} h \tag{13}$$

[1] actually the computation of the other points would not be useful, since the two points and the marker direction determine the tip position.

Where h is a given parameter of the pencil model, equal to the distance from the mean of the two low points of the marker to the projection of the pencil tip on the marker plane.
Let d be another given parameter of the pencil model, representing the distance between the pencil tip and the marker plane. To obtain the pencil tip position, it is sufficient to move along the plane normal of a distance d from $\mathbf{X}_t^{c'}$, in one of the two possible directions. The presence two directions is the effect of the vertical symmetry. We call $\mathbf{n}_+$ the true one, that satisfies:

$$\mathbf{n}_+ \cdot \mathbf{k} = \mathbf{n}_+ \begin{pmatrix} 0 & 0 & 1 \end{pmatrix} > 0 \tag{14}$$

thus the one increasing the dept in the camera frame, since the marker is visible only when $\mathbf{X}_t^{c'}$ is at lower depth than $\mathbf{X}_t^c$.
In the end the pencil tip 3D position in the camera frame is:

$$\mathbf{X}_t^c = \mathbf{X}_t^{c'} + d\mathbf{n}_+ \tag{15}$$

6 Trajectory Smoothing and Outliers Rejection

This stage of our algorithm is a minor but necessary stage.
The reconstruction is done frame by frame, so the output of the reconstruction of one frame is not correlated with the neighboring frames. If the detection phase is inaccurate reconstructed position can be very far from its neighborhood.
During this stage we tackle this problem by considering the sequential nature of the drawing.
Given a vector of reconstructed 3D points we apply smoothing for each coordinate independently using a sliding window of a fixed size. For each window we calculate the mean $\mathbf{X}_{mean}$ and the standard deviation $\mathbf{X}_{std}$ along each coordinate. We label point i in the window as outlier if its distance from the mean is greater than a threshold dependent on the standard deviation of the window. In other words point i is an outlier if:

$$\|\mathbf{X}_i - \mathbf{X}_{mean}\| \geq k * \mathbf{X}_{std}\,. \tag{16}$$

After having removed outliers in the window we calculate the new mean and substitute the reconstructed positions with the mean position.
In our settings $k = 1.5$ and the window size is 10. This parameters can be tuned on the specific video.

7 Experimental Results

In this section we present the result of the application of our algorithm on two videos.

7.1 First Video

In this video we draw the word "ciao" in italic letters. Here we do not raise up the pencil too much from the table but the drawing is fast. We provide the link to the full video [2]. We first apply the crossratio reconstruction method (section 5.2), whose results are shown in fig. 6. The reconstruction is quite accurate and the letters are visible. We compare the results of our extension of

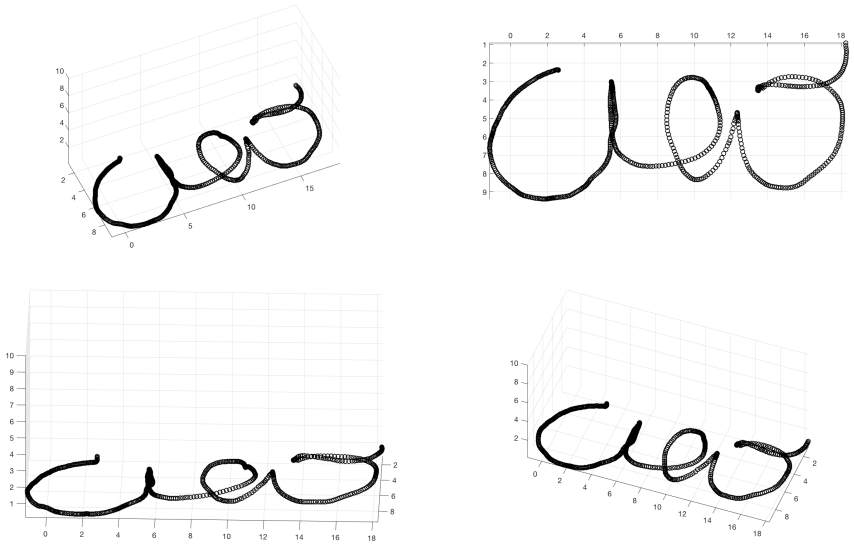


Figure 6: Different views of the reconstruction of section 7.1 using the crossratio method.

the three collinear point reconstruction method with the ones given by the trivial solution (Section section 5.1). Using this method the obtained results are not satisfactory (see fig. 7). There's only one view (that corresponds approximately with camera view) in which the reconstruction looks good.

7.2 Second Video

In this video we draw the word "ciao" in capital letters. From one letter to another we raise up the pencil to test the correctness of our algorithm on the height. Here we provide the link to the full video [3].

[2] `https://youtu.be/dky71el55qo`
[3] `https://youtu.be/U7XAzXeBx-U`

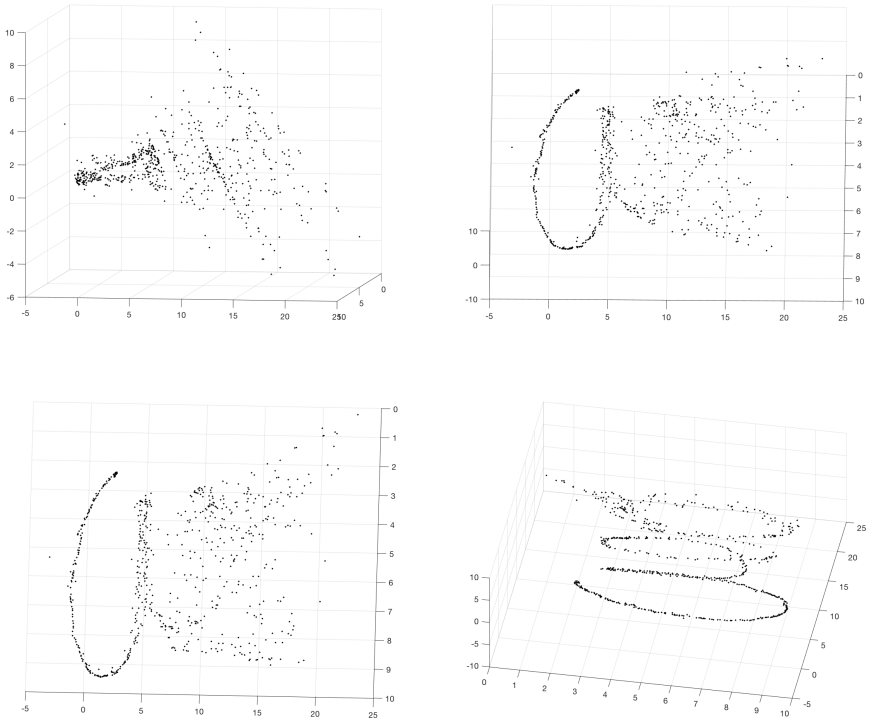


Figure 7: Different views of the reconstruction of section 7.1 using the homography reconstruction method.

In fig. 8 we can see different views of the result. Based on a comparison with the original video we can say that the reconstruction is satisfactory. The word "ciao" drawn during the video is well distinguishable and the frames with the pencil at an higher z are detected.

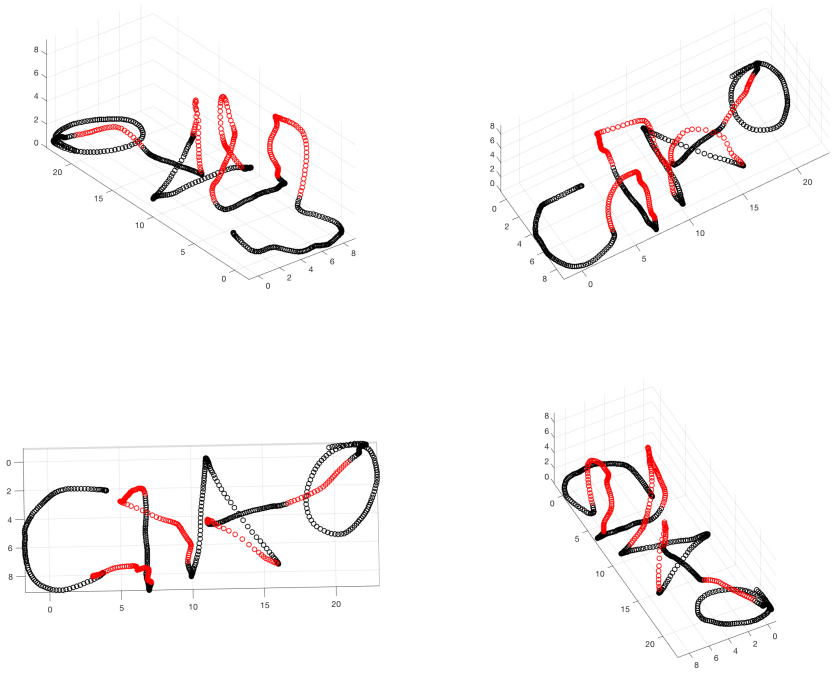


Figure 8: Reconstruction of video in section 7.2 with crossratio. Different views of the result, points are red when their z coordinate is greater than 1.5 cm

8 Future work

In this section we describe some improvements that can be inserted in our pipeline to make our writing tool more robust or more user friendly.

8.1 Detection improvements

The detection step is based on a user provided RGB filter. This means that many videos of the same pen taken in different conditions need different filters. At the expense of complexity, two different solutions can be implemented to make the pencil marker detection step more robust. Both are based on finding in the current frame a bounding box around the pencil marker, in order to apply the successive detection steps on a robustly obtained region.

RANSAC for homography Instead of printing the pencil marker on a white sheet, we can print it on a textured background, similar for instance to the calibration marker. Given a template of the pencil marker, we can match template features to current frame features (SURF, for instance) and then fit an homography that brings template points in frame points. The application of such a homography to the corners coordinates of the template gives the corner of the pencil marker in the current frame.
The successive step could be, instead of looking for the lines in the current frame, to transform the pencil marker lines in the template using the fitted homography.
Note that this way of proceeding is valid only if the marker is attached to a planar surface.

Neural network Machine learning techniques are widely employed in vision problems. For object detection, a common approach is the one of transfer learning, that consists in the reuse of an already trained convolutional neural network. The procedure is the following: the last neurons of the pre-trained network are substituted with other kind of neurons, depending on what kind of output function we need, and the resulting network is fine-tuned for the object detection task.
The drawback of this method is that a dataset need to be created. For robustness with respect different points of view, pencils, and light conditions, the dataset need to contain samples of a wide set of cases.

8.2 Pencil marker improvements

Our pencil marker presents two major limitations: it can be partially or totally occluded by the pencil, and it presents a vertical symmetry (thus the need of the assumption that the pencil is always directed toward the bottom).

Occlusion This limitation can be overcomed by substituting our marker with a similar one, that is attached around the whole pen, and presents many vertical lines in such a way that two or more of them are always visible. The detection algorithm should take in account this, in different ways:

- Keeping only a pair of near vertical lines.
- Keeping all the lines. Adaptiveness to the number of visible lines is needed in the reconstruction step, and the radius of the pencil need to be known to build a marker model that is no more planar when the number of visible lines is greater than three.

The detection step could also use the Hough algorithm for finding ellipses instead of lines, to detect the horizontal elements of the marker.

Vertical symmetry The marker should be enhanced with distinctive visual information in its intersection points, recognized in the detection step. This would allow to solve the data association problem in the vertical direction, and understand when the pencil tip is not directed toward the writing surface

References

Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Comput. Vis. Image Underst.*, 110(3):346–359, June 2008.

Duane C. Brown. Close-range camera calibration. *PHOTOGRAMMETRIC ENGINEERING*, 37(8):855–866, 1971.

B. Caprile and V. Torre. Using vanishing points for camera calibration. *Int. J. Comput. Vision*, 4(2):127–140, May 1990.

Martin A. Fischler and Robert C. Bolles. Random sample consensus: A paradigm for model fitting with applications to image analysis and automated cartography. *Commun. ACM*, 24(6):381–395, June 1981.

Richard Hartley and Andrew Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003.

Richard I. Hartley. Self-calibration from multiple views with a rotating camera. In Gerhard Goos, Juris Hartmanis, and Jan-Olof Eklundh, editors, *Computer Vision — ECCV '94*, volume 800, pages 471–478. Springer Berlin Heidelberg, Berlin, Heidelberg, 1994.

Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am. A*, 4(4):629–642, Apr 1987.

Kenneth Levenberg. A method for the solution of certain non-linear problems in least squares. *Quarterly of Applied Mathematics*, 2(2):164–168, jul 1944.

Donald W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *SIAM Journal on Applied Mathematics*, 11(2):431–441, 1963.

Stephen J. Maybank and Olivier D. Faugeras. A theory of self-calibration of a moving camera. *International Journal of Computer Vision*, 8(2):123–151, Aug 1992.

Zhengyou Zhang. *A flexible new technique for camera calibration*. 1998.

A Planar object localization

A planar object is determined by a rotation matrix $\mathbf{R}$ and a translation vector $\mathbf{t}$ expressing its rotation and its position in the camera reference frame. Given an image of a known planar scene a point on the plane $[X\,Y\,0\,1]^t$ is projected on the image plane using the following relationship:

$$\begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = s\mathbf{K}\,[\mathbf{R} \mid \mathbf{t}] \begin{bmatrix} X \\ Y \\ 0 \\ 1 \end{bmatrix} \tag{17}$$

$$= s\mathbf{K} \begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} \tag{18}$$

$$= \mathbf{H} \begin{bmatrix} X \\ Y \\ 1 \end{bmatrix} . \tag{19}$$

Using equations 18 and 19 we have:

$$\begin{bmatrix} \mathbf{r}_1 & \mathbf{r}_2 & \mathbf{t} \end{bmatrix} = \mathbf{K}^{-1}\mathbf{H} . \tag{20}$$

Then $\mathbf{r}_3$ can be written as vector product of $\mathbf{r}_1$ and $\mathbf{r}_2$.
The resulting rotation matrix might not be orthogonal, so we make it orthogonal using the following procedure:

$$[\mathbf{U}, \mathbf{V}] = svd(\tilde{\mathbf{R}}) \tag{21}$$

$$\mathbf{R} = \mathbf{U}\mathbf{V}^T . \tag{22}$$

B Three collinear points reconstruction

Given the image, taken with a *calibrated camera* ($\mathbf{P} = [\mathbf{M} \mid m]$ is known), of three collinear points whose real world distances are known, this algorithm reconstructs their 3D position in the camera frame.

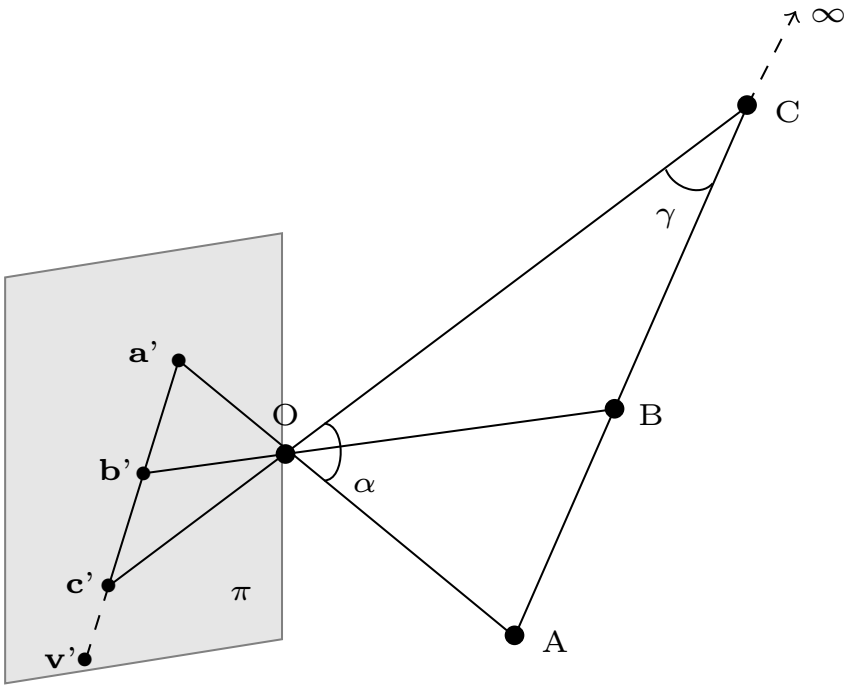


Figure 9: Illustration of reconstruction of three collinear points

In fig. 9 we have $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ that are points in the real world, $\mathbf{a}'$, $\mathbf{b}'$, $\mathbf{c}'$ that are their images and $\mathbf{v}'$ that is the image of the vanishing point of the direction identified by the line passing through $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$. The point $\mathbf{O}$ is the camera center, the plane π is the image plane. The object modeled by the three points it is completely defined by its orientation and its position. The point $\mathbf{v}'$ can be determined by imposing the cross ratio invariance:

$$\frac{\|\mathbf{a}' - \mathbf{c}'\| \, \|\mathbf{b}' - \mathbf{v}'\|}{\|\mathbf{b}' - \mathbf{c}'\| \, \|\mathbf{a}' - \mathbf{v}'\|} = \frac{\|\mathbf{A} - \mathbf{C}\|}{\|\mathbf{B} - \mathbf{C}\|} \,. \tag{23}$$

This is a non-linear scalar equation. We need at least two constraints for determining $\mathbf{v}'$, so we impose that $\mathbf{v}'$ belongs to the line $\mathbf{l}$ joining $\mathbf{a}'$, $\mathbf{b}'$, $\mathbf{c}'$:

$$\mathbf{l}^t \mathbf{v}' = 0 \,. \tag{24}$$

Orientation The orientation of the object is the orientation of the line passing through $\mathbf{A}, \mathbf{B}, \mathbf{C}$. This can be easily determined as the back projection ray of $\mathbf{v}'$, i.e. the line passing through $\mathbf{v}'$ and $\mathbf{O}$, with equation $\mathbf{M}^{-1}\mathbf{v}'$.

Position The position of the object in the camera frame is the distance from $\mathbf{O}$ to $\mathbf{A}$. We can apply the sine theorem to the triangle $\overset{\triangle}{\mathbf{ABC}}$:

$$\overline{\mathbf{OA}} = \overline{\mathbf{AC}} \frac{sin(\gamma)}{sin(\alpha)} \,. \tag{25}$$

The angle α is easily determined as the angle between lines $\mathbf{M}^{-1}\mathbf{v}'$ and $\mathbf{M}^{-1}\mathbf{a}'$:

$$cos(\alpha) = \frac{(\mathbf{M}^{-1}\mathbf{v}')^t(\mathbf{M}^{-1}\mathbf{a}')}{|| \mathbf{M}^{-1}\mathbf{v}' ||_2 || \mathbf{M}^{-1}\mathbf{a}' ||_2} \,. \tag{26}$$

Similarly, γ is the angle between lines $\mathbf{M}^{-1}\mathbf{v}'$ and $\mathbf{M}^{-1}\mathbf{c}'$.

C Absolute orientation

Suppose that we are given the coordinates of points as measured in two different reference frames. The problem of recovering the transformation between these two from the points coordinates is known as absolute orientation.
Formally, given correspondences $\{(\mathbf{X}_i^A, \mathbf{X}_i^B)\}_{i=1..N}$, we seek for the solution of the following minimization problem:

$$\begin{aligned} \min_{s,\mathbf{R},\mathbf{t}} & \sum_{i=1}^{N} \left\| s\mathbf{R}\mathbf{X}_i^A + \mathbf{t} - \mathbf{X}_i^B \right\|^2 \\ s.t. \quad & \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \in \mathbf{SE}(3) \\ & s > 0 \ . \end{aligned} \tag{27}$$

Note that, if the model of an object is known and described by a set of 3D points, this algorithm can be used to fit the model to a noisy perception of the object. It is sufficient to find the transformation that best aligns the object model to the current perception, and then apply it to the object model. The result is the least square model in the perception reference frame.
We employ the Horn's closed form solution (Horn, 1987), without taking in account the scale difference s, not present in our application.
Here we report the solution to the minimization problem:

$$\begin{aligned} \min_{\mathbf{R},\mathbf{t}} & \sum_{i=1}^{N} \left\| \mathbf{R}\mathbf{X}_i^A + \mathbf{t} - \mathbf{X}_i^B \right\|^2 \\ s.t. \quad & \begin{bmatrix} \mathbf{R} & \mathbf{t} \\ \mathbf{0}^T & 1 \end{bmatrix} \in \mathbf{SE}(3) \ . \end{aligned} \tag{28}$$

The reader is referred to the original paper for a full explanation.

Rotation Let $\bar{\mathbf{X}}^A$ and $\bar{\mathbf{X}}^B$ be the centroids of the two set of points, and $\bar{\mathbf{X}}_i^F = \mathbf{X}_i^F - \bar{\mathbf{X}}^F$ for $F \in \{A, B\}$. Let $\mathbf{M}$ be the matrix of sum of products of coordinates measured in the A system with coordinates measured in the B system:

$$\mathbf{M} = \sum_{i=1}^{N} \bar{\mathbf{X}}_i^A \cdot \bar{\mathbf{X}}_i^B = \begin{bmatrix} S_{xx} & S_{xy} & S_{xz} \\ S_{yx} & S_{yy} & S_{yz} \\ S_{zx} & S_{zy} & S_{zz} \end{bmatrix} \ . \tag{29}$$

The quaternion representing the best rotation is the eigenvector associated to the highest eigenvalue of the matrix:

$$\mathbf{S} = \begin{bmatrix} S_{xx} + S_{yy} + S_{zz} & S_{yz} - S_{zy} & S_{zx} - S_{xz} & S_{xy} - S_{yx} \\ S_{yz} - S_{zy} & S_{xx} - S_{yy} - S_{zz} & S_{xy} + S_{yx} & S_{zx} + S_{xz} \\ S_{yz} - S_{zy} & S_{xy} + S_{yx} & -S_{xx} + S_{yy} - S_{zz} & S_{yx} + S_{xy} \\ S_{xy} - S_{yx} & S_{zx} + S_{xz} & S_{yx} + S_{xy} & -S_{xx} - S_{yy} + S_{zz} \end{bmatrix} \tag{30}$$

whose elements are sum and differences of the elements of $\mathbf{M}$.

Translation Let $\mathbf{R}$ rotation matrix corresponding to the best quaternion, the translation is obtained as:

$$\mathbf{t} = \bar{\mathbf{X}}^B - \mathbf{R}\bar{\mathbf{X}}^A \ . \tag{31}$$

D Dilation and Erosion

Binary images may contain numerous imperfections. In particular, the binary regions produced by simple thresholding are distorted by noise and texture. Morphological image processing pursues the goals of removing these imperfections by accounting for the form and structure of the image. Morphological techniques probe an image with a small shape or template called a *structuring element*. The structuring element is positioned at all possible locations in the image and it is compared with the corresponding neighborhood of pixels. The structuring element is a small binary image, i.e. a small matrix of pixels, each with a value of zero or one. The structuring element is said to **fit** the image if, for each of its pixels set to 1, the corresponding image pixel is also 1. Similarly, a structuring element is said to **hit**, or intersect, an image if, at least for one of its pixels set to 1 the corresponding image pixel is also 1.

Erosion The erosion of a binary image f by a structuring element s produces a new binary image g with ones in all locations (x, y) of a structuring element's origin at which that structuring element s fits the input image and 0 otherwise, repeating for all pixel coordinates (x, y). Erosion has the effect of shrinking white regions. Erosion with small square structuring elements shrinks an image by stripping away a layer of pixels from both the inner and outer boundaries of regions. The holes and gaps between different regions become larger, and small details are eliminated.

Dilation The dilation of an image f by a structuring element s produces a new binary image g with ones in all locations (x, y) of a structuring element's orogin at which that structuring element s hits the the input image f and 0 otherwise, repeating for all pixel coordinates (x, y). Dilation has the opposite effect to erosion, it adds a layer of pixels to both the inner and outer boundaries of regions.