Master's Degree programme – Second Cycle
(D.M. 270/2004) in Computer Science

Final Thesis

Università
Ca'Foscari
Venezia

Robust Joint Selection of Camera Orientations and Feature Projections over Multiple Views

Supervisor:
Ch. Prof. Andrea Albarelli

Ca' Foscari
Dorsoduro 3246
30123 Venezia

Graduand:
Mara Pistellato
Matriculation Number 839976

Academic Year
2015/2016

Contents

Abstract

A number of critical factors arises when a complex 3D scene has to be reconstructed by means of a large sequence of different views. Some of them are related to the ability of identifying the projection of each observed 3D point. Others are tied to the reliability of the pose estimate of each view. In particular, accurate intrinsic and extrinsic calibration of capturing devices is a crucial factor in any reconstruction scenario. This task becomes even more problematic when we are dealing with a large number of cameras, in fact we can be unable to observe the same object from all the viewpoints or we can suffer from unavoidable displacement of cameras over time.
With this thesis we propose a method which tries to solve these problems at the same time, while also be inherently resilient to outliers. We propose a game-theoretical method that can be used to simultaneously select the most reliable rigid motion between cameras together with the projections on the image planes of the same 3d feature, which position in 3D space is recovered by means of triangulation.
The original inception then has been further refined to address a wider range of scenarios, as well as to offer a reduced memory consumption and computation complexity. By exploiting these enhancements, we were able to apply this technique to a large scale setup involving several hundreds of view points and tens of thousands of independent observations.

Chapter 1

Introduction

The main purpose of Computer Vision is the attempt to extract information and knowledge from images using various, heterogeneous tools that come from Mathematics, Physics, Computer Science and so on. Such techniques can be applied to a wide range of different scenarios, involving both practical and theoretical issues. To this end, it is very difficult to produce an unique definition for this discipline since it is very multifaceted. We can say that, in general, Computer Vision's principal goal involves the interpretation of the world by automatic image analysis. The tools needed in order to reach such goal involve imaging hardware for acquiring and storing data, algorithms for processing the images and several precautions that must be adopted to make these activities reliable and effective.
These methods are designed with the purpose of letting a computer perform some tasks that come natural for an human: immediate examples are image feature recognition and segmentation.

Whenever a problem is addressed using Computer Vision methods, the main hurdle that arises is the loss of information which will inevitably result from the imaging process itself. Such loss of information is both semantic, as objects from the real world are translated to intensities values on a discrete imaging plane, and syntactic, as projective geometry involves an inherent dimensionality reduction and spatial deformation.
The main goal of any method proposed in Computer Vision literature is indeed to recover from these kind of hindrances in order to fit a model as accurate as possible with respect to the real world and to extract some semantics from it.
A simple example is background separation from a sequence of frames [52], where the objective is to recognize which pixels belong to the background and to separate them from the rest (this is often a preliminary step to per-

form further activities). From a human point of view this problem is trivial since our cognitive system can immediately recognize the scene and give a meaning to it. Differently, this results to be a dunting task to be addressed in a perfectly reliable manner by an automated system. Even with a good stable camera and providing a perfect illumination, this simple task is subject to large inaccuracies and lack of repeatability that have been only partially addressed by the several approaches proposed in literature during the last decades.

The problem addressed with this thesis is in the field of multiview 3D reconstruction. This kind of structure recovery problem includes at least two sub-tasks. The former one is directly related to the reconstruction of 3D geometry from 2D projections, the latter aims at finding the correct correspondences between different projections of the same 3D entity across several different cameras.
3D geometry recovery tackles the fundamental problem of estimating the depth of an observed material point with respect to the capturing camera. In fact, such information has been completely lost during the imaging process and a special technique must be adopted to recover it.
Some of these techniques are able to work with only one camera, still, all such approaches are usually difficult to apply and exhibit a rather low accuracy. The process becomes a lot easier if two or more cameras are available, since a wide range of triangulation methods can be exploited to reconstruct the scene. For such operation to be feasible, however, the correspondence between features observed from different points of view must be known, and this happens to be a difficult task by itself.
This thesis introduces a novel a method to solve both steps at once. The method is general enough to work with any number of cameras and can be applied also to dynamic networks.

1.1 The imaging process

The imaging formation process [57] consists in capturing through a camera a portion of a scene. The goal of the process is to get intensity images, which are the familiar pictures that we are used to see; these images are 2D arrays which encode light intensity acquired by cameras.
Intensity images measure the amount of light which hit a photosensitive sensor and encode the information as an array of numbers.

In human visual system, the light rays that come from outside hit the photoreceptors in the retina, which will transmit the intensity information to the brain. In the digital image formation we have the same principle.
There are a variety of physical parameters involved in image formation, like the optical parameters of the lens which characterize the sensor's optics; the photometric parameters which model the light energy reaching the sensor after being reflected by object in the scene; and geometric parameters which determine where a 3D point is projected.

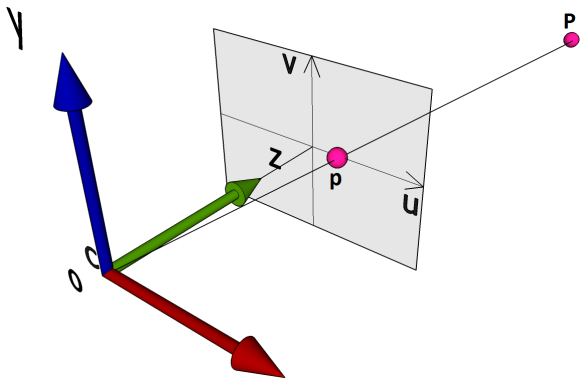

Figure 1.1: Simple pinhole camera model. Here we have that the origin of the Euclidean axes of the world reference frame is aligned to the camera reference frame. Centre of focus **C** corresponds to origin **O**.

The most simple geometric model of an intensity camera is the pinhole model [24]. This model is formed by an image plane and a 3D point **C** called the centre of projection. In the simplest case **C** corresponds to the origin **O** of the Euclidean coordinate system which is the origin in the world coordinate system.

Light rays in the the real world are emitted in many directions, consequently they can be reflected by physical objects in many directions. In pinhole model the rays enter from a single aperture point and then they are projected on the image plane, which is orthogonal with the z axis.
Since for each physical point we have only one ray that enters the camera, we have a one-to-one correspondence between visible points and image points. In this way all the points at any distance are in focus in the image plane.
The distance between the centre of projection **C** and the image plane is called focal length f and the perpendicular line which connects them is the optical axis. The optical axis meets the image plane in the principal point.

A simple representation with an aligned reference system is shown in figure 1.1.

Exposure time of a pinhole camera is rather long since it allows only few light to enter through the aperture, so it needs more time in order to register the image. For this reason real cameras are much more complex: they include a lens system which allow to have a much larger aperture (instead of a really small one) and a shorter exposure time.

We now move to the geometric aspect of image formation. In this part we are interested in how the position of a scene point is connected with its corresponding image point. In the following part we analyse the model of geometric projection performed by the sensor.

The most common geometric model is the perspective or pinhole model that we already described. As shown in figure 1.1, $\mathbf{p}$ is the point at which the line through $\mathbf{O}$ and the correspondent real point $\mathbf{P}$ intersects the image plane.
Let $\mathbf{P} = (X, Y, Z)^T$ and $\mathbf{p} = (x, y, z)^T$ in the 3D reference frame in which $\mathbf{O}$ is the origin. Then, the two points are linked by the following fundamental equations

$$x = f\frac{X}{Z} \quad ; \qquad y = f\frac{Y}{Z} \tag{1.1}$$

Since the image plane is orthogonal to the z axis, its equation is $z = f$. All image points lay on the image plane, so the third component of the image point $\mathbf{p}$ is always equal to the focal length. The coordinates of $\mathbf{p}$ in fact can be written as $\mathbf{p} = (x, y, f)^T$.

The above relations work in one camera reference frame, but most of the time we have that the camera reference model is located with respect to some other reference frame, called world reference frame. Moreover, in a digital image only the pixel coordinates are directly available, so we need a method to obtain the image points in the camera reference frame. These issues can be modelled if we assume the knowledge of some characteristics of the camera: these are known as intrinsic and extrinsic parameters.

Intrinsic parameters are values needed to compute pixel coordinates of an image point from the corresponding coordinates in the 3D scene, expressed in the camera reference frame.
Extrinsic parameters define the location and orientation of the camera refer-

ence frame with respect to a known world reference frame of another camera in case we have more than one.

In the following part we define the basic equations that allow us to define extrinsic and intrinsic parameters. The problem of estimating the values of these parameters is called camera calibration and we will discus about it later.

Intrinsic Parameters

Intrinsic parameters are defined as the set of values needed to parametrize the characteristic imaging model of the viewing camera.

Recalling the pinhole camera model, we defined the principal point as the point in which the centre of projection **C** meets the image plane in a perpendicular way. In real cameras this point is rarely aligned with precision, so we need two parameters to encode the information of its displacement: these values are indicated as c_x and c_y.

The optics in real cameras often introduce image distortion, which is more evident at the periphery of the image. Distortion can be modelled rather accurately as a radial distortion with the following equations

$$\begin{aligned} x &= x_d(1 + k_1 r^2 + k_2 r^4) \\ y &= y_d(1 + k_1 r^2 + k_2 r^4) \end{aligned} \tag{1.2}$$

Where (x_d, y_d) are the coordinates of the distorted point and $r^2 = x_d^2 + y_d^2$. The distortion is a radial displacement of the image points: it is zero at image centre and increases as the points are further from the centre. k_1 and k_2 are two additional intrinsic parameters and are usually very small. In the following equations we do not consider distortion parameter since they are ignored when high accuracy is not required in all regions of the image. Moreover, distortion can always be removed by applying the inverse of equations 1.2 to the captured image.

Once a 3D point is projected through the ideal pinhole, we must transform its coordinates according to the relative position of the sensor with respect to the origin.

For this purpose we can define the camera calibration matrix K, which contains the intrinsic or internal parameters of the camera.

$$K = \begin{pmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{pmatrix} \tag{1.3}$$

While f is the focal length as defined before, here we introduce f_x and f_y because we assume independents focal lengths in x and y dimensions.

In order to estimate the intrinsic parameters of a camera, we could use some targets with known geometry [63] and take different pictures of the object in different positions. By identifying the target features in every image we are able to compute the intrinsic parameters by solving a linear system.

Extrinsic Parameters

The camera reference frame is often unknown, and we have a common reference frame called world reference frame. Figure 1.2 shows this situation. A frequent problem is determining the location and orientation of the camera frame with respect to some known world reference frame.

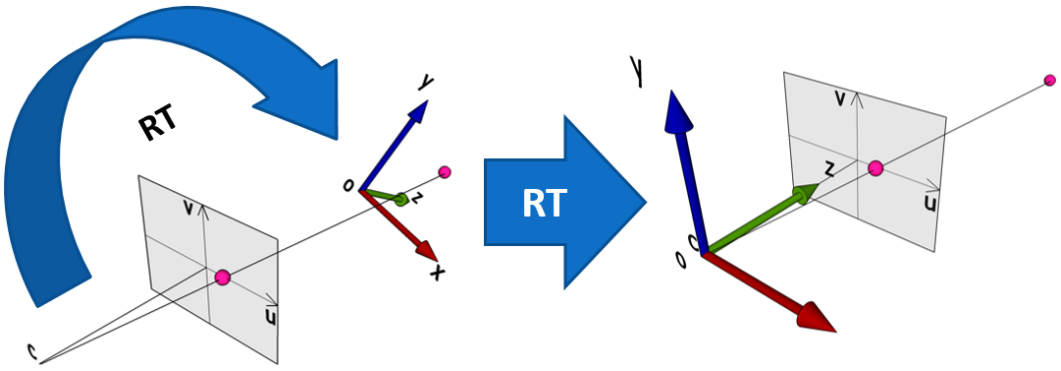

Figure 1.2: The world reference frame does not correspond with the camera reference frame (left). Rotation and translation information are needed in order to align the reference frames.

Extrinsic parameters are defined as any set of parameters that can uniquely identify the transformation between the unknown camera reference frame and

a known reference frame. Typically the transformation is described using two factors: translation and rotation.
A translation is represented as a 3D vector $\mathbf{T}$ which describes the relative position of the origin of the two reference frames. Rotation is expressed as a 3×3 matrix R, an orthogonal matrix that contains the rotation which aligns the corresponding axes of the two frames.
Recall the observed point $\mathbf{P}$ and let $\mathbf{P}_c$ and $\mathbf{P}_w$ to be its coordinates respectively in camera reference frame and world reference frame. Their relation is

$$\mathbf{P}_c = R(\mathbf{P}_w - \mathbf{T}) \tag{1.4}$$

We can combine the extrinsic parameters in a 3×4 roto translation matrix RT just adding vector $\mathbf{T}$ to the matrix R as the fourth column.

$$RT = [R|\mathbf{T}] \tag{1.5}$$

Projection and Homogeneous Coordinates

Points in 3D are often expressed in homogeneous coordinates [24]. We add an extra coordinate to each point and represent such points by equivalence classes of coordinates.
A point in a 2-dimensional space (x, y) can be represented by the equivalence class (kx, ky, k) with $k \neq 0$. For example the point $(2x, 2y, 2)$ in homogeneous coordinates is equivalent to $(x, y, 1)$. If we want to get the original coordinates from a point (kx, ky, k), we just need to divide each of its components by k.
In computer vision, projective space is used as a convenient way of representing the real 3D world, extending the 3-dimensional space. As we discussed before, images are formed by projecting the world into a 2-dimensional representation and so they are conveniently extended to be lying in the 2-dimensional projective space. This homogeneous vector representation for points will be used in our projective space.

Projection is simply a map from the 3-dimensional projective space into the 2-dimensional projective space. If points are expressed as homogeneous vectors, the mapping from a real world point to the corresponding point in the image plane is obtained multiplying the 4-dimensional point vector with the projection matrix P, defined as follows

$$P = K * RT \tag{1.6}$$

We have that matrix RT performs the transformation between the world and the camera reference, while K performs the transformation between the camera reference and the image reference frame.
If the world point is represented with the vector $\mathbf{X} = (X, Y, Z, 1)^T$, its projection on the image plane is

$$\mathbf{x} = P\mathbf{X} \tag{1.7}$$

The resulting $\mathbf{x}$ is a homogeneous 3-dimensional vector $(x, y, z)^T$. Since it is homogeneous, we can compute its equivalent $(x/z, y/z, 1)$ and the ratios x/z and y/z are exactly the image coordinates.
The relation between $\mathbf{X}$ and $\mathbf{x}$ can be seen as a linear transformation from the 3D real world projective space to the projective plane. The transformation is defined up to an arbitrary scale factor.

Once we have the formed image, we would like to filter the noise, extract some semantic content from it or to operate on a reduced dimensionality.
Various approaches have been formulated and various techniques have been proposed for each particular issue. In the following sections we describe some known basic image processing tasks that can be performed as an initial step in order to solve Computer Vision problems.

1.1.1 Feature Detection

In Computer Vision, the term image feature can refer to a global property of an image (for instance the average grey level, the number of pixels and so on) or to a part of the image with some special properties. In fact, given an image we can extract some particular features from it: lines, edges, corners, circles, other shapes and so on [55]. We can be particularly interested in extracting lines and corners, since these elements are not distorted by the image formation process and often they have a semantic meaning. If, for example, we have a sphere to be detected in our scene, it will always be seen as a circle no matter the angle from which we capture it.
Local image features should be meaningful: they must be associated with interesting scene elements. Moreover they must be detectable, in such a way that several algorithms which are able to extract them exist and are feasible.

Edge detection

Edges in an image are points where intensity changes in a very rapid way: usually we are interested in studying them because they correspond to objects boundaries or other relevant image features. It is hard to define what

changes we want to extract because we need high-level information to tell if an edge is semantically relevant or not.

We can treat the image as a function $I(x, y)$ which values are the recorded intensity values. Given a function, the first derivative of it expresses the slope in a certain point, so it is computed in order to extract the points where rapid changes happen, which corresponds to high values.
Since $I(x, y)$ is a discrete function in two variables, its slope and growth direction can be computed as the gradient [13]

$$\nabla I = \left(\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y} \right) \tag{1.8}$$

Which is a vector that contains both partial derivatives of the image I. The partial discrete derivative in x direction is defined as

$$\frac{\partial I}{\partial x} \approx I(i+1, j) - I(i-1, j) \tag{1.9}$$

Computing this partial derivative for each pixel of the image is the same as convolving the image using the kernel

$$\begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & -1 \\ 0 & 0 & 0 \end{pmatrix} \tag{1.10}$$

If we transpose the given kernel we can compute the partial derivative in vertical direction.

If we compute the gradient magnitude (its norm) in each point of the image, we will have a measure of how likely that point is part of an edge, since the gradient magnitude will be higher where we have a rapid change of light. In other words, we have a large gradient magnitude associated to sharp changes in the image. In figure 1.3 we can see an example of the gradient magnitude extracted from a grayscale image.
Once we have the gradient magnitude for each pixel, we can apply a threshold in order to extract the final border points.

The gradient approach is actually the most simple (but very effective) one for edge detection. Many techniques have been proposed in order to extract edges from images: some use different filter like Laplacian of a Gaussian [38],

Figure 1.3: Gradient magnitude values of an image. Note that it is stronger in points where we have a large change in intensity in a particular direction.

other adopt different approaches [45].
Moreover, the theory of edge detection can be applied in a very wide range of fields wherever a reliable image processing approach is needed, for example in astronomy [58].

Lines and Shapes detection

Once we have some edges, we could be interested in extracting the straight lines from them, maybe because we know we are observing a geometric object, like a chessboard, and we are interested in extracting the grid lines.
For example, markers with a known geometric pattern are widely used in order to calibrate cameras [63] or to detect and track some objects in the scene [32].

One method to perform this task is using the Hough Transform, which is a simple technique used to find lines but it could be easily generalized for finding different shapes like ellipses [17] [7]. An example of application is shown in figure 1.4, where we applied the following algorithm to figure 1.3.

In this approach we represent each line in a parametric space (d, θ) where d is the distance from the line to the origin (or the radius) and θ is the angle between that segment and the x axis. As a consequence of this parametriza-

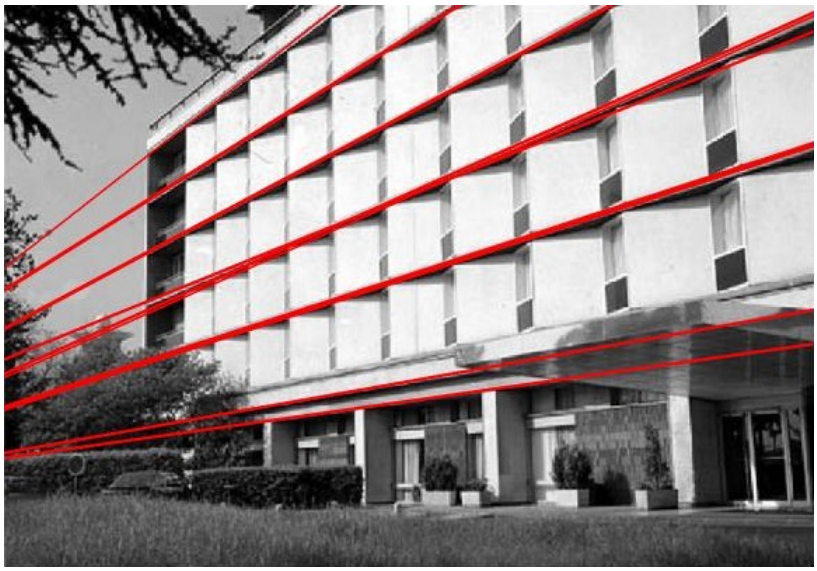

Figure 1.4: Lines extracted using the Hough transform starting from borders extracted in figure 1.3.

tion, we have that a line in the original image corresponds to a point in the transformed Hough space.
On the other hand, a border point (x_0, y_0) in spatial coordinates can be crossed by an infinite number of lines, so the family of straight lines passing through (x_0, y_0) becomes a sinusoid in the Hough space, because it includes all possible couples (d, θ) that can pass through that point. The relation between an image point (x_0, y_0) and the sinusoid in Hough space is given by the following transform

$$d = x_0 cos\theta + y_0 sin\theta \tag{1.11}$$

The algorithm takes each border point and represents it in the Hough space, giving a vote for each possible (d, θ). An example of how the Hough space looks like is given by figure 1.5. The parameters which received more votes are classified as lines in the original picture.

We can extend this method in order to find more shapes. In particular, we can have a multidimensional parametric space, where each dimension represents a parameter: if we want to find ellipses, we must use five parameters, so instead of lines we obtain 5-dimensional curve surfaces to intersect in order to get the most voted ellipses.

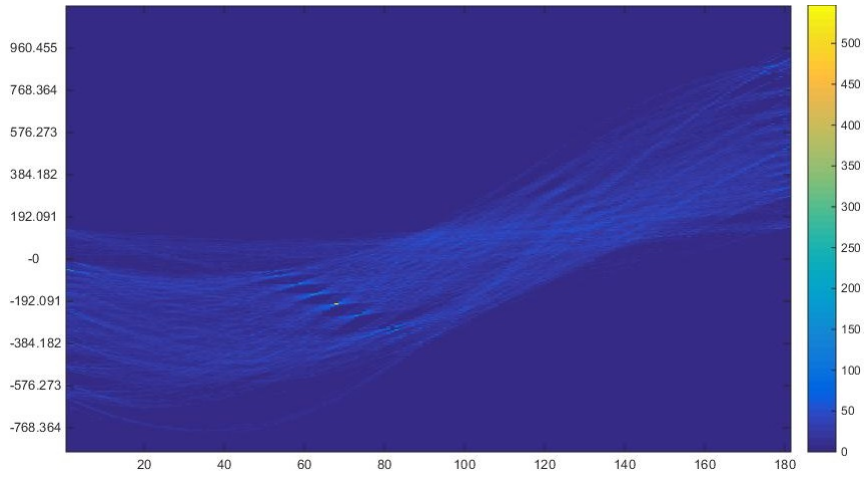

Figure 1.5: Example of an instance of Hough space: note that the sinusoids tend to intersect in particular points that will be the detected lines.

1.1.2 Image Segmentation

Image segmentation is the task of dividing a given image into regions which are semantically separated [55]. The goal is to identify groups of pixels that can be grouped together in such a way that they represents different objects in the scene. We can have many ways to define such regions: for example we can define a similarity measure and search for regions with high intra-region consistency.

A classical approach is Normalized Cut [50], where the affinity between two pixels is defined in terms of distance and colour similarity. The image is represented as a weighted graph where each pixel is a node and on which we define an affinity matrix $A = (a_{ij})$. The element a_{ij} represents the weight between nodes i and j, which corresponds to the affinity function between pixel i and pixel j.
When the complete graph is divided in two parts, a cut is created: it can be seen as a partition which creates two sets of pixels from the original image. The cut value is then defined as the sum of all the weights of edges that cross the cut (which are edges that connect the two groups).
The goal of the algorithm is to find a partition that minimizes the cut value in such a way that we have the maximum affinity inside both the divided groups. Once we have the minimum weight cut, we can apply the algorithm recursively in both parts in order to find additional partitions.

Of course we can define the affinity between pixels in several ways according to brightness, colour, texture or motion information in addition to simple proximity. The general affinity function is defined with a Gaussian kernel

$$aff(i,j) = \begin{cases} e^{\frac{-(F(i)-F(j))^2}{\sigma_I^2} + \frac{-d(X(i),X(j))^2}{\sigma_X^2}} & \text{if } d(i,j) < r \\ 0 & \text{otherwise} \end{cases} \tag{1.12}$$

Where σ_I and σ_X are the scales for respectively the image feature and point spatial distance and $d(x,y)$ is the spatial distance between the two points. The value r is the maximum distance to which two nodes are allowed to be connected by an edge. For each pixel we have $X(i)$ which is the spatial location and the value $F(i)$, which is a feature vector which can represent several characteristics of that pixel.

If a segmentation based only on brightness is required we have that $F(i)$ is simply the intensity value of pixel i; if we want a segmentation based on colour, $F(i)$ can be the colour vector, represented in some particular colour space like HSV [35], which is designed for such tasks. Another option is to have $F(i)$ as some filters at various scales and orientations, in case of texture segmentation.
Figure 1.6 shows the original and segmented image obtained applying normalized cut based on distance and colour similarity between pixels.

Figure 1.6: Example of image segmentation performed with normalized cut algorithm.

Other techniques have been developed for image segmentation, based on graphs [19] or more dynamic ones, based on region growing techniques called snake models [64] [62].

1.1.3 Background Subtraction

Given a sequence of frames which are usually part of a video, background subtraction techniques try to divide the background from the foreground of the images, in order to extract the relevant objects in the scene.
If we have a recording of an highway for example, we wish to extract only the image portions which depict the cars, since we are not interested in the street itself. Another common example is human body action recognition [40]: we put a camera in a room and we want to detect only the people moving inside of it, then once people are detected further processing can be done in order to recover limbs for example.
This is a very difficult task because we could have fast changes in image luminosity and of course the system may not know a priori which elements are to be considered foreground. Moreover, we could have a repetitive motion in the background or elements which change in very long time.

The simplest models we can think are based on mean or median filters: in these approaches the background model is built in each pixel to be the mean or the median of previous N pixels in the same position.
These approaches are simple and fast, moreover they change the model during time, which is what we want, but the accuracy actually depends on the velocity of objects and on frame rate: if we have very fast objects with respect to the frame rate we have no good results; also we could have change in luminosity as said before.

More complex techniques have been proposed in order to address this task; some of them are based on a probabilistic model which try to adapt itself by learning on-line some distribution parameters which better adapt to background pixels [52] [30] [29]. In this approaches we use a mixture of K Gaussians for each pixel in order to learn the background model: at each step we can adjust the means and the standard deviations of the distributions in order to fit the background colour. The distributions are ordered by some weights and their variance: the smaller is the variance and the higher is the weight, the more probability we have that the distribution represents a background pixel.

The probability of a pixel X_t observed at time t to be in the background is

$$P(X_t) = \sum_{i=1}^{B} \omega_i \mathcal{N}(X_t|\mu_i, \Sigma_i) \tag{1.13}$$

Where ω_i, μ_i and Σ_i are respectively the weight, mean and covariance matrix of the i-th Gaussian. B is the number of Gaussians selected to represent the background model.

1.2 Recovering the Scene Structure

The focus is now put on the spatial properties that can be reconstructed from acquired images. Until now the images have been examined as they are presented, like 2-dimensional items. In previous section the interest was only in detecting some relevant features from images like lines, edges or particular objects.
Now some possible approaches that can be exploited to recover the full three-dimensional scene information are analysed.
We already discussed that when an intensity image is captured (e.g. only the intensity of light is measured), all the information about the spatial location of the observed items is lost. In other words, the depth of the scene elements is unknown, consequently methods to recover this information are needed.
This loss leads to a paradox, known as reconstruction paradox: actually an external observer is not able to tell if a real 3D object is captured by a camera or if the same camera is capturing a simple picture of the same real object, which technically is just a plane item in the real world. The image formation process in fact produces the same result but the camera is not observing the same thing. This paradox is well depicted in figure 1.7.

More ambiguities like proportions have also to be faced: objects near to the camera seem to be bigger than objects which are far away. For an human brain is simple to recognize the correct dimensions but a computer system can not tell because it has no spatial or semantic information.
In this and the following sections different solutions for the 3D reconstruction and location problem are analysed. Section 1.3 puts the focus on multi-view approach, since it is the principal subject of this thesis.

If other fields beyond simple intensity images are explored, more sophisticated tools can be used in order to recover information about spatial depth of observed objects in the scene. This is the case of range images, which can

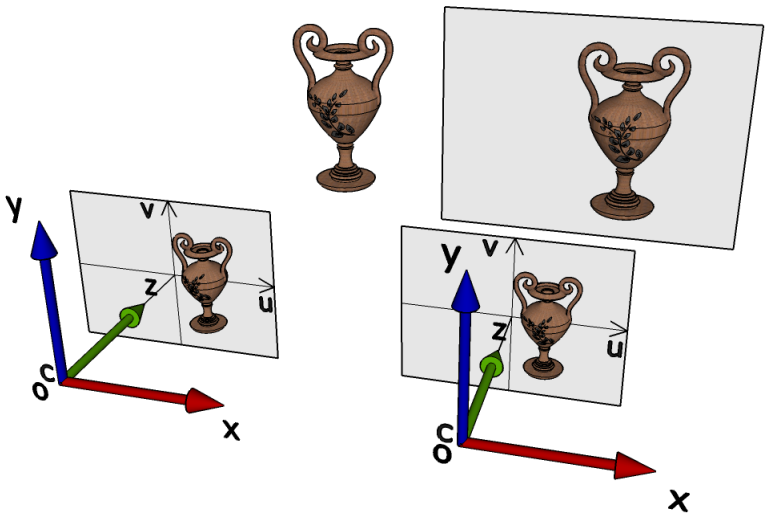

Figure 1.7: Reconstruction paradox: the two cameras produce the same image but one comes from a real vase, the other from a picture of the same vase. Looking at the captured images, a machine is not able to tell which one is from a real vase and which is not.

encodes shape and distance information of the scene, rather than intensity of light.
These images are acquired using special sensors like sonars or laser scanners: these devices are able to directly acquire the 3D structure of the environment. Of course, several specific Computer Vision techniques have been developed in order to work with this kind of images [15] [16].

1.2.1 3D Reconstruction

Algorithms that aim to reconstruct the 3D structure of a scene or to locate a point in 3D space need a set of equations to link the actual 3D points in space with the coordinates of their corresponding captured image points [24] [57]. These equations need to know the characteristics of the cameras and their parameters are intrinsic and extrinsic parameters.
Even knowing all camera parameters, when we detect a 2D point on image plane we have an infinite set of 3D points which could match that observation. In other words, if we want to solve the inverse of equation 1.7 given the 2D point $\mathbf{x}$ we find infinite solutions which all lie on the line that starts in the centre of projection and passes through point $\mathbf{x}$. Figure 1.8 is a graphical representation of this typical situation.

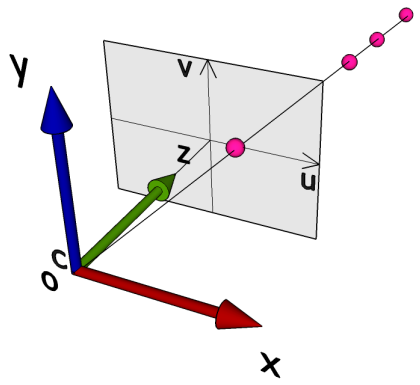

Figure 1.8: Possible 3D points given a 2D point on image plane.

When these problems are considered, two main approaches could be adopted: in the single view approach one single camera is exploited in order to reconstruct the scene; in the second approach two or more cameras are used in a combined way in order to locate the items. In the following part single view techniques are discussed and then in the next section multiple view techniques are explained.

In a single camera configuration different approaches have been proposed, in which the aim is to recover the depth position of a given image point.

Depth From Defocus

One possible solution which involves a single camera system is called Depth From Defocus [54] [53].
In order to obtain sharp images, all rays coming from the same scene point must converge to a single point on image plane: in this case we say that the point is on focus. If a point is not on focus the image is spread over a disc centred on that point.
As said before, in practice cameras are not pinhole: in this basic model we have only one plane which is perfectly on focus because only one ray from a given point enters the camera. A real camera uses lenses instead, in order to change the focus plane. As a consequence, when we change the focus plane working with lenses, we can focus on points at different levels of depth from the camera image plane.

In this approach $A \times F$ pictures are captured, with A different apertures $\{\alpha_1, ..., \alpha_A\}$ and F focus settings $\{f_1, ..., f_F\}$. The goal is to compute for each pixel its in-focus setting. The correct focus value is found when the aperture value of camera is not relevant. Once the correct focus plane is recovered for a pixel, the depth of that pixel can be easily computed.
This method uses only one camera and could be made more automatic employing some specialized hardware. On the other hand, it has several shortcomings: a high number of pictures of the same still scene are needed, so this method can not be used with some moving objects. Moreover, highly textured objects are needed because if the scene includes an item with only plain colours (like a completely white statue) the system is not able to tell which parts are in focus and which are not. Also, very good hardware is required. For these reasons this technique is not very used since it leads most of the time to poor results.

Other single camera techniques are actually combined solutions which use a kind of additional sensor to the classical camera.

Time Of Flight

In general, time of flight describes a variety of methods that measure the time needed for an object to travel a distance through a medium. In this particular approach light is used.

Time-of-flight camera is a range imaging camera system that computes distance based on the known speed of light, measuring the time of flight of a light signal between the camera and the subject.
In this configuration a light source is mounted near the camera. This source emits a pulsed signal and then measures the time needed for the reflection of the light signal to come back; since the overall time depends directly on the distance between the emitter and the object hit, it is possible to compute the spatial distance.
This method is extremely fast and could easily be used in a wide range of real-time applications [14] [21]. A practical example are car safety applications [18].
In general this approach produces reasonably dense features and it is supported by most of hardware. There are also a lot of drawbacks: with particularly small objects it is not very accurate and could produce a lot of noise, moreover it works poorly on dark or distant objects so it is not suitable where we require a precise and robust method.

Laser Scanning

Other approaches involve the use of a laser plane to detect the position of surfaces [34] [61].
Laser is known to be very accurate so that the reconstruction is feasible and very precise but requires a lot of images. Figure 1.9 shows a simple example of the system configuration.
Different setups can be proposed, for example the object could be put on a turntable and then it is reconstructed by joining all the cilindric profiles. Another technique consists in a fixed object and a moving laser light.

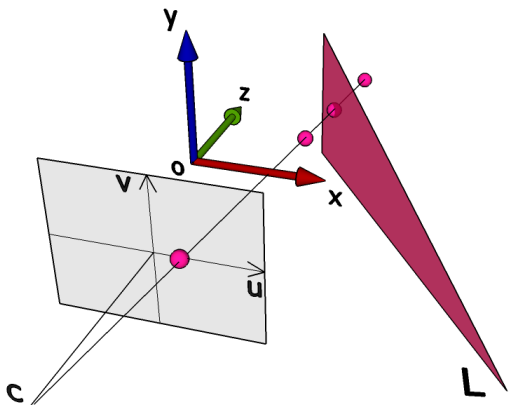

Figure 1.9: Laser scanning process involves a laser plane emitted from a source L which is calibrated with the camera.

1.3 Multiple View Reconstruction

Stereo vision is referred to the ability to infer on the 3D structure and distance of a scene from two or more images taken from different viewpoints.
A stereo system has to solve two problems:

- Correspondence: is the problem of determining which item from one view corresponds to which in the other view;
- Reconstruction: conversion of the observed points to a 3D map of the viewed scene.

The way in which a stereo system determines the position of a point in space is triangulation: the rays defined by centres of projection and the images of the correspondent points are intersected and the 3D coordinates are recovered. Of course triangulation depends highly on the correspondence problem.
In order to have a working stereo system to perform triangulation, calibration of some parameters is needed: the intrinsic parameters for each camera which characterize the transformation mapping an image point from camera to pixel coordinates, and the extrinsic parameters which describe the relative position and orientation of the two cameras.

In a basic multiview setting, two cameras can see the same portion of the scene. This assumption is quite strict and not plausible, in fact in real applications there are many cases with occlusions or partial coverage of the scene in one or more cameras.
Assume for now that both cameras are able to see the same object: first there is the need to find the correspondence between the two observations.
Correspondences could be found in different ways, for example using structured light patterns projected directly on the objects or exploiting directly the scene features.

1.3.1 Triangulation

Once the correspondences between the pair of images are available and if both extrinsic and intrinsic parameters are known, the 3D reconstruction problem can be solved by easily compute the 3D coordinates of the feature with respect to a common world reference by means of triangulation.

In Geometry, triangulation identifies the process of computing the third point coordinates of a triangle knowing the other two points and the two angles.
In Computer Vision the problem is analogous, in fact it is defined as the process of determining a point position in the 3D world starting from its images in two cameras and from the camera parameters [24] [57]. Figure 1.10 is a graphical representation of the scenario.

Let point $\mathbf{P}$ be the point projected in both camera image plans and its projections $\mathbf{p}_l$ and $\mathbf{p}_r$. Note that $\mathbf{P}$ lies at the intersection of the two rays from $\mathbf{O}_l$ and $\mathbf{O}_r$ that pass respectively through $\mathbf{p}_l$ and $\mathbf{p}_r$.

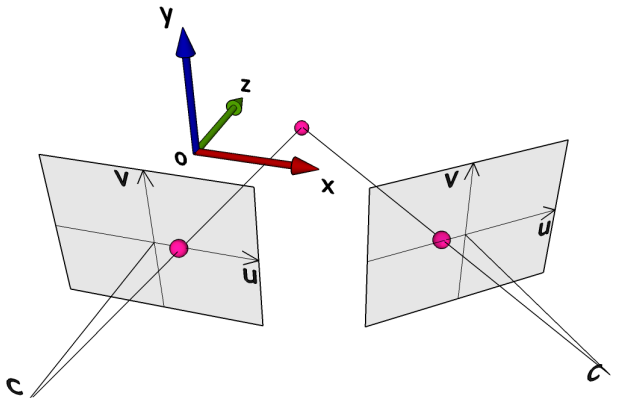

Figure 1.10: Process of triangulation: both cameras detect the same point and compute its 3D coordinates.

Since rays are known, the intersection could be computed, but the camera parameters and points position are approximated measures so the two rays do not really intersect in space. We can have only an estimation of the point position by computing the point of minimum distance from both rays.

We express all vectors and coordinates referred to the left camera reference frame and define the equations for both the rays. We denote $a\mathbf{p}_l$ with $a \in \mathbb{R}$ as ray l through $\mathbf{O}_r$ (case $a = 0$) and $\mathbf{p}_l$ (case $a = 1$). Let also $\mathbf{T} + bR^T\mathbf{p}_r$ with $b \in \mathbb{R}$ be the ray r with $\mathbf{p}_r$ expressed in the left camera reference frame.
The vector orthogonal to both l and r is $\mathbf{w} = \mathbf{p}_l \times R^T\mathbf{p}_r$. We call w the line through $a\mathbf{p}_l$ and parallel to vector $\mathbf{w}$: its equation is $a\mathbf{p}_l + c\mathbf{w}$ with $c \in \mathbb{R}$.

If we fix two values a_0 and b_0, the endpoints of the segment from $a_0\mathbf{p}_l$ to $\mathbf{T} + b_0R^T\mathbf{p}_r$ can be easily computed solving the linear system of equations

$$a\mathbf{p}_l - bR^T\mathbf{p}_r + c(\mathbf{p}_l \times R^T\mathbf{p}_r) = \mathbf{T} \tag{1.14}$$

In the same way we can define the segment s belonging to the line parallel to $\mathbf{w}$ that joins l and r. We can determine the endpoints of s in the same way and the triangulated point $\mathbf{P}'$ is the midpoint of segment s. The system has a unique solution if and only if the two rays l and r are not parallel.

1.3.2 Camera networks

In general, two points of view are enough to reconstruct 3D information from 2D projections. Nevertheless, in many practical scenarios the adoption of multiple independent cameras to acquire the scene is the best choice in order to reduce errors. In fact, in most of real-world applications we would like to perform an accurate tracking or reconstruction.
This is the case, for instance, when people are to be tracked in large areas for security reasons [31] and strong resilience to occlusion is required.
A collection of different points of view can also result from dynamic scenarios, where cameras are mounted on drones [26] or images are collected by different users on social networks or online services [1].
Camera networks can also be helpful when the phenomenon we want to study is complex and difficult to analyse from just one point of view. We have many computer vision applications where this is required: human action recognition [48], video surveillance [59] and object tracking [41].
The adoption of multiple cameras could finally lead to improved accuracy with image-based surface reconstruction [20] especially when dealing with complex artefacts [6].

As stated before, image based 3D reconstruction relies on two principal factors: the ability to match observations from different cameras and the knowledge of the relative pose between cameras.
Both problems have been widely analysed over the last decades, and a large number of solutions have been proposed.

1.3.3 Calibration and Pose Estimation

For any of these applications to be feasible we first need to perform intrinsic and extrinsic calibration: the geometry of cameras must be known, at least with a given precision. This is a fundamental step if we want to initialize our system in such a way that it works well.

A lot of calibration methods that can be used to recover intrinsic and extrinsic parameters have been proposed. Classical approaches use artificial targets with known geometry [63] to compute intrinsic parameters and simultaneously assess the relative pose of each camera [25]. These methods give an initial permanent configuration of the system. We also have dynamic methods, which consist in adapting camera parameters and structure reconstruction exploiting some scene features that could be artificial markers or simply particular objects [36].

Other methods perform pairwise calibrations that can be made consistent with respect to a common reference world [44] [56]. Other techniques propose to calibrate a whole camera network at the same time [8].

Regardless the chosen method, any calibration procedure will always result in some degree of inaccuracy. In addition, even very accurate calibrations could deteriorate over time due to external environmental factors or as consequence of camera movements we could have slight changes in the network topology or drift of intrinsic parameters.

Some pose estimators adopt special calibration targets characterized by a known geometrical model such as squares and circles. Such approaches are only feasible when dealing with fixed cameras that can be calibrated offline in a preparation phase.
Self calibration methods allow to exploit directly the observed features in order to compute the parameters. Usually these methods minimize the overall reprojection error of feature points triangulated under the estimated poses. Self calibration is particularly useful when addressing scenarios including multiple cameras organized in a network or a sequence of frames generated by an unknown camera motion.

When dealing with self calibration, points labelling and pose estimation are tightly correlated tasks so we would introduce uncertainty on point localization and wrong feature labelling: these errors lead to a large pose estimation error because the function that we want to minimize has wrong assumptions.
An inaccurate pose estimation could block any attempt to evaluate correspondences. Many solutions have been proposed to deal with this problem.

1.3.4 Correspondences

Structured light

The basic idea of structured light approach is to project some light patterns directly onto the scene we want to reconstruct, then each camera has to recognize and label each point in the same way so that at the end of the process point correspondences between all viewpoints are provided and can be triangulated.
In the naive approach just one small point at time is projected on the scene and all cameras register that point with the same identifier. The process goes on until all possible points have been projected. This approach works but

it is very expensive in terms of time, in fact even with a very fast system, it can take hours to register all possible points allowed by projector resolution.

A more efficient technique consists in creating some codes made by the light patterns. In figure 1.11 a simple code is shown: we assume that light codes a one and the absence of light codes a zero.
Initially we project stripes of an alternate 0/1 pattern with the maximal resolution available. Each camera records the black and white points and the correspondent value is saved as the less significant bit in that point coding information.
In the following steps exactly half stripes than before are projected, and cameras code the zeros and ones as the second less significant bit. The process continues until the last pattern is formed by only two parts. At the end of this coding phase, each point in each view will have a unique identifier and couple of points with the same identifier can be triangulated to recover the 3D structure of the scene. Figure 1.11 shows the sequence of projected patterns.

Figure 1.11: Sequences of binary patterns projected on the scene in structured light approach. Black codes a zero and white codes a one.

Further improvements can be obtained if the number of patterns to be projected is reduced: we can do this by increasing the number of different intensity levels in the stripes encoding. Examples could be stripes with multilevel grey colours instead of binary, or coloured stripes [27].
Many techniques have been proposed in order to achieve more accuracy or to solve some known problem which can be related to projection, for example in the presence of surface points which are not reached by the light source [4].

The advantages that can be achieved with this method is that we have an high accuracy in recovering the pixel correspondences, the procedure is also

faster compared with the laser scanning approach. Moreover different patterns could be chosen according to the scene we want to reconstruct: some patterns could be more suitable than others and a calibrated light source is not needed like in other approaches.

There are also some shortcomings: the performances are not good with shiny objects like metals which reflect the light, it is not suitable for large areas and the technique can not be used if we already have an images collection from which we have to extract the 3D structure, for example a sequence of pictures collected by cameras mounted on a structure.

Image Features

Another kind of approach is to extract some image features directly by processing the pictures. Techniques described in previous sections could be used to extract some image features and then a similarity function between them could be defined in order to detect the possible correspondences.

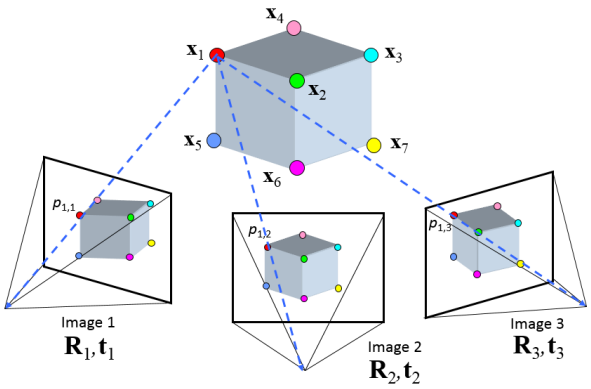

Figure 1.12: Feature matching on edges of a cube with different colours.

To obtain a good feature, it must satisfy two properties: repeatability and distinctiveness. Repeatability assigns that the feature should be available in different views and its position should be well recognizable. Distinctiveness ensures that each feature descriptor can be clearly distinct from another if the material point is not the same.

A simple method is to extract image corners [23] and analyse surrounding pixels. This approach has several problems, since the quality of corners is really connected to the kind of image.

If we have some particular, geometrical features corners offer a good match, but in other common situations the corner extractor does not find only the real corners, but simply points in which the intensity changes in all the directions.

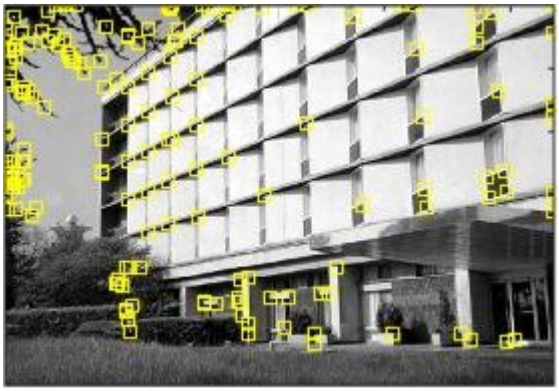

Figure 1.13: Corners extracted with Harris detector: some corners from the building are correct but many corners form the tree are not actual corners.

A more sophisticated approach consists in local descriptors. In fact correspondences between images can be found by exploiting the local appearance of the scene, by means of descriptors which are able to capture such information.

SIFT (Scale Invariant Feature Transform) [37] is the most common. It offers invariance with respect to scale, rotation, illumination and in some cases also to deformation. The image content is transformed into local feature coordinates.
The algorithm that produces SIFT descriptors involves many steps: first it runs a linear filter (difference of Gaussians) at different resolutions of the same image in order to find the corners at different scale level.
An efficient function is to compute the Difference of Gaussian pyramid [12]. Then, some corner keypoints are extracted and thresholded.
In order to ensure invariance to orientation for each feature, local orientation is computed by finding the strongest second derivative direction, then each element is rotated so that orientation points up.
Figure 1.14 shows a practical example of how SIFT feature descriptors can be used to match image features when we are dealing with the same image with a different scale and rotation.

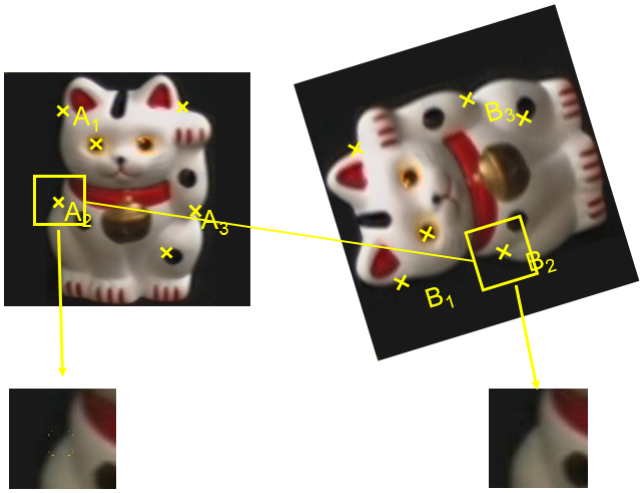

Figure 1.14: SIFT feature descriptors allow to recognise the same image feature when we have different rotation and scale.

Many similar techniques have been proposed, as SURF [9], GLOH [39], BRISK [33], FREAK [2] and some others.

This kind of approach works with existing photographic material, unlike structured light, and can be use in a very wide range of problems.
Some applications of local descriptors include: composing panorama picture from SIFT descriptors correspondences [10], unsupervised object recognition [11], 3D scene reconstruction and many others.

All these descriptors try to ensure good repeatability and distinctiveness properties but they are still prone to lead to false matching due to noise or similarity in appearance of objects.
Correspondence reliability could be improved using high-level matching frameworks accounting for multi-feature consistency or by discarding photometric descriptors substituting them with more robust identification methods including structured light or artificial markers, where it is possible.

Chapter 2

Selection of Orientations and Features

In this thesis we propose a consensus-based approach to select the most reliable set of extrinsic parameters that can be used to perform triangulation and feature selection given the existing calibration. The key idea of the consensus approach is to adopt a game-theoretical validation of mutually consistent observations so that we obtain a simultaneous selection of camera orientations and corresponding feature projections over the network.

In order to do this we do not propose a new calibration technique, and we are neither interested in how the network calibration has been performed; we assume that a possibly large set of cameras is available and that some previous process assessed their extrinsic positions up to its best accuracy.
Our goal is not to enhance the existing calibration or to correct the precision, but to select the best set of poses and features.

Our initial approach consists in selecting paths of rigid transforms connecting cameras observing the same physical point and choosing the paths such that the triangulation error between every couple of cameras is minimized.
After that, a further optimized method is proposed: it operates directly on the point images and not on their triangulations as before, reducing the number of total hypothesis and improving the performances. This enhanced approach permits to work on unlabelled points and to find correspondences between observations. Finally, if we are able to compute a compatibility function between observations, it can be used to reduce probability of mismatch.
The general approach is dynamic and depends on several factors like the po-

sitions of material points, the quality of observed features and of course the quality of initial extrinsic estimation.

In what follows intrinsic camera calibration refers to all parameters needed to convert the image acquired by cameras to normalized image plane as captured by an ideal pinhole imaging process.
Differently, the term extrinsic calibration is used to define the rigid transformations relating the camera reference frame to a common world frame.

2.1 General Scenario Description

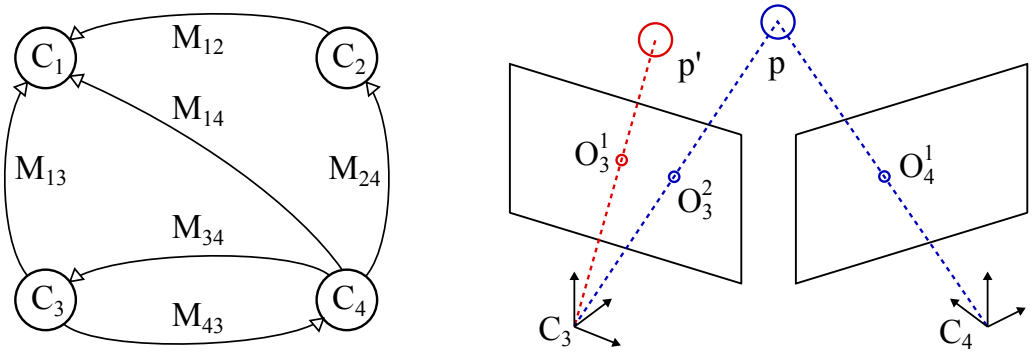

Figure 2.1: Overview of scenario. Multiple cameras are nodes of a partially incomplete graph of extrinsic transformation. A physical point is observed by several cameras with a random positional noise and a possibly wrong labelling.

The general scenario we are working with is well represented in figure 2.1, where a small version of extrinsic graph $G = (C, M)$ is represented.

We are dealing with a network of cameras $C = \{C_1, C_2, ..., C_n\}$ each referred by a unique label C_i, with $i = 1..n$.
The size of this network can range from a few to several tens of independent devices and they can be located in the same restricted area or spread in a wider environment.
As said before, we assume that cameras have been previously calibrated and that we already have intrinsic and extrinsic parameters.

Extrinsic calibration process has been modelled as a set of rigid transformations $M = \{M_{11}, M_{12}, ..., M_{nn}\}$ where M_{ij} is a 4×4 roto-translation matrix which transforms points expressed in the reference world of C_j to a point expressed in the reference frame of C_i.
In this model we assume $M_{ii} = I$ but we are not assuming other matrices to be transitively consistent, in fact the calibration method that produced set M is considered to be pairwise so in general we have that $M_{ij} \neq M_{ji}$ because we have no assumptions on calibration process. Consequently we have that $M_{ij} \neq M_{ik}M_{kj}$ and so on for every possible concatenation of the matrices.

If the calibration method is global all elements in M would be transitively consistent but we have no guarantees on the accuracy obtained since it depends on the construction process. For example if they are built by incrementally chaining pairwise calibrations we have that errors easily sum up. Usually global methods perform some averaging process or try to minimize errors computing a priori optimal paths in the given graph. Our method proposes the opposite approach: we keep the inconsistent graph G and then we select the optimal weighted subset of paths from $\mathcal{P}(M)$.

We assume two or more cameras in C are able to observe the same physical point, but each camera could see only a subset of the points in the scene. That will result in a number of points in image planes which are affected by a random observation noise modelled as a Gaussian distribution. Moreover, since we can have several features in the scene, the labelling process that permits to identify the same feature in different cameras can fail, leading to have wrong labelled features. Finally, the rigid transform between a pair of cameras might not be available and of course could be subject to uncertainty.

In the following we will refer to the observation sequentially labelled a in the camera C_i as O_i^a. In figure 2.1 (right) we can see an example: cameras C_3 and C_4 observed material point p and a single observation labelled O_4^1 is reported in camera C_4. On the other hand, camera C_3 reports two observations, which are O_3^2 and O_3^1: one comes from the correct point p while the other comes from the wrong labelling of p' in which we are not interested. The goal of proposed method is to select the best combination of camera orientations and point projections to get the most reliable 3D reconstruction.

2.2 Game Theoretical Optimal Path Selection

Game Theory [43] was introduced in the 40's by J. Von Neumann to model the behaviour of entities with competing objectives. The original aim was to create formal and simple description of the strategies adopted in economic fields such as company competitions or consumer decisions using a mathematical model characterized by a single objective function.
The theory was further developed by John Nash in postwar period through the introduction of the Nash Equilibrium [42].

During the years, Game Theory had also been successfully applied in different fields as Biology [22], animal behaviour studies, social sciences [51], Psychology, Philosophy, Computer Science and Logic.

With the Nash equilibrium, the emphasis shifts from the search of an optimum shared by the population to the definition of equilibria between opposite forces. The main intuition behind Game Theory is that we can model a competitive behaviour between agents - or players - as a game where a finite number of predefined strategies are available and there is a fixed payoff gained by a player when all the other agents choose their strategies to play.

Evolutionary Game Theory [60] considers a scenario where pairs of individuals, each programmed with a strategy, are repeatedly drawn from a large population to play a game. A selection process allows the best strategies to grow while the others are driven to extinction.
The underlying idea of using Evolutionary Game Theory for selection is to model each hypothesis as a strategy and let them be played one against the other until a stable population emerges.

The basic assumption of Optimal Path Selection (OPS) [46] is that a mostly correct labelling already exists, so correspondences are already established and the optimal path can be selected independently for each point. We will see that the hypotheses are made up of all possible triangulations, including two paths and two observations.

Normal Form Game

A normal form game is composed of a set of players, also called agents $I = \{1, 2, ..., n\}$. Each player has a finite set of actions $S_i = \{1, 2, ..., m_i\}$ where m_i is the number of possible actions for player i and S_i is the set of

actions of player i. Such actions are called pure strategies.
A pure strategy profile is defined as $S = S_1 \times S_2 \times ... \times S_n$.

A payoff function is defined as

$$\pi : S \to \mathbb{R}^n; \qquad \pi(s) = (\pi_1(s), \pi_2(s), ..., \pi_n(s)) \tag{2.1}$$

Where π_i is the payoff function of player i. In other words, it is the utility that player i gets when the pure strategy profile s is played.

If a simultaneous-move game is assumed, all players at a certain time play one of their strategies with no knowledge of the other players' strategies. Once all actions have been decided, each player gets its own profit according to its payoff function. The purpose of each player in the game is to maximize its own payoff function, but of course the payoff depends on all other strategies. In this way we build a system in which each agent is against all the others. These games are also called non-cooperative since players can not form any coalitions against the others.

We want now to define the behaviour of each player: as in real world, there are strategies which are more common and others which are rarely adopted. To model this fact mixed strategies are defined.
A mixed strategy x_i for player i is a probability distribution over the set of its strategies S_i. It is a point in the standard simplex

$$\Delta_i = \{x_i \in \mathbb{R}^{m_i} : \forall h = 1..m_i x_{ih} \geq 0 \text{ and } \sum_{h=1}^{m_i} x_{ih} = 1\} \tag{2.2}$$

x_i is a m_i-dimensional vector where x_{ih} is the probability that player i plays its pure strategy h. The support $\sigma(x_i)$ is the set of pure strategies of player i with a positive probability. As we did with the pure strategies, we can put together all the mixed strategies from all players and obtain a mixed strategy profile $x = (x_1, x_2, ..., x_n)$ which lays in the multi-simplex space $\Theta = \Delta_1 \times \Delta_2 \times ... \times \Delta_n$. Note that corners of the standard simplex correspond to pure strategy because we have only one point different to zero: so mixed strategies can be seen as a generalization of pure strategies.

Evolutionary Games

The conflict can be seen as a two players, symmetric game, in which players behave according to their genetic pattern which is pre-programmed. Since

each player always plays one predefined strategy, mixed strategies are interpreted as the fraction of the population which play a strategy. In other words, in evolutionary interpretation the individuals play a fixed strategy but the whole population is partitioned in groups that play strategies according to the mixed strategy.
We have a symmetric game with two players, this means that we do not distinguish between player 1 and player 2 since they can be exchanged. The game can be modelled with a symmetric matrix A, called payoff matrix. Element a_{ij} corresponds to payoff value when one player plays strategy i and the other plays strategy j.
In this case we can apply the Fundamental Theorem of Natural Selection which states that for any doubly symmetric game, the average population payoff $x^T Ax$ is strictly increasing along any non-constant trajectory of replicator dynamics.
The ideal goal is to reach an evolutionary stable strategy, which is a mixed strategy resilient to invasion by new populations.
In order to simulate the evolutionary process we can apply replicator dynamics. The key idea in this approach is that when we have a strategy that behaves better than the average, it is going to spread and if we have a strategy worst of the average, it is going to disappear.

In order to adapt our approach to an evolutionary game definition, the set of hypotheses is defined according to all possible camera paths and observations. Moreover, a payoff function is designed in order to represent the mutual support of two hypothesis. The fundamental idea is that if two of our hypotheses "play well" together, then they can be part of the final population selected by the replicator dynamic process.

2.2.1 Hypotheses

We define $H = \{H_1, H_2, ..., H_k\}$ to be the set of all hypotheses. Each one is a possible triangulation and includes two observations and the two paths connecting the observing cameras to the world frame that we use as global reference. We can assume that the world frame is aligned with C_1 without loss of generality.

Each hypothesis is a quadruple $(M_{1x}...M_{yi}, O_i^a, M_{1w}...M_{vj}, O_j^b)$ where $M_{1x}...M_{yi}$ and $M_{1w}...M_{vj}$ are paths that combine a sequence of rigid transformations which transport respectively a point from camera C_i in camera C_1 frame reference and an observation from camera C_j to camera C_1. As already stated, the accuracy of these transformations depend on the calibration algorithm.

Observations O_i^a and O_j^b are hopefully two observations of the same physical point observed from C_i and C_j.

Each hypothesis must contain two different cameras so that in previous definition we have $i \neq j$; also the paths must not include cycles and are shorter than a maximum length *maxpath* defined as parameter.
The correctness of each triangulation in an hypothesis depends on many factors which can be related to either calibration process or observation labelling: we could have a misclassified point or Gaussian errors from calibration and image acquisition. For these reasons in general it is impossible to state if an hypothesis is a valid candidate or not.

2.2.2 Payoff Function

As any hypothesis alone is completely non informative, we have to focus on the definition of how two hypotheses support each other. This measure is called payoff and it should be high if the two hypotheses support the same 3D point and low if they are discordant.
The payoff is defined as a real-valued function $\pi(i,j) : H \times H \rightarrow \mathbb{R}^+$ where i and j are identifiers of hypotheses H_i and H_j. Since we can define a payoff value for each pair of hypotheses, we can collect all values in a squared payoff matrix $\Pi = (\pi_{ij})$ with $\pi_{ij} = \pi(i,j)$.
From each hypothesis H_i it is possible to obtain an associated 3D point $x(H_i)$ through triangulation. The technique used for triangulation does not affect our method since we add in the evaluation of the payoff the skewness value $s(H_i)$ which corresponds to the minimum distance between the two rays used to recovered the point $x(H_i)$. Coordinates of recovered points and skewness values from both hypotheses are taken in account to compute the payoff value. Two hypotheses are considered compatible is their triangulated 3D points are close: the closest the points are, the more compatible the two hypotheses will be. Each pair can also contribute to the overall reliability measure using the skewness in the same way. To simplify the model we can consider these two measures as independent and approximate the similarity between two hypothesis with a bidimensional Gaussian kernel:

$$\pi'(i,j) = e^{-\frac{1}{2}\left(\frac{(|x(H_i)-x(H_j)|)^2}{\sigma_p^2}+\frac{max(s(H_i),s(H_j))^2}{\sigma_s^2}\right)} \tag{2.3}$$

Where σ_p and σ_s are two parameters that represent respectively the expected standard deviation of points position and of the skewness.

Note that $(|x(H_i) - x(H_j)|)^2$ is a pairwise measure that needs both hypotheses H_i and H_j to be computed; while the skewness measures $s(H_i)$ and $s(H_j)$ are computed independently one from another so that to put them together in the pairwise function π' we need the max operator. We will exploit this skewness property in the following improvement so that we will not need to triangulate every pair of observations but still obtain the same results.

While π' function expresses the degree of consensus between any two hypotheses, we need to account for special cases where two hypotheses are not compatible regardless the quality of the triangulation. For example, when two hypotheses include two different observations from the same camera, triangulate them is pointless because we are considering two different points for sure. Another infeasible case consists in the presence of two different paths from the same camera: if we assume that for different points we can have different optimal paths, this is not the case of two different paths for the same point on the same camera since it would break the common world constraint. These special cases could be included explicitly in the payoff function setting the value to zero in the final payoff.

$$\pi(H_i, H_j) = \pi((P^u_\alpha, O^a_\alpha, P^{u'}_{\alpha'}, O^{a'}_{\alpha'}), (P^v_\beta, O^b_\beta, P^{v'}_{\beta'}, O^{b'}_{\beta'})) =$$

$$= \begin{cases} 0 & \text{if } \alpha = \beta \wedge (u \neq v \vee a \neq b), \\ & \alpha' = \beta' \wedge (u' \neq v' \vee a' \neq b'), \\ & \alpha' = \beta \wedge (u' \neq v \vee a' \neq b), \\ & \alpha = \beta' \wedge (u \neq v' \vee a \neq b'). \\ \pi'(i,j) & \text{otherwise} \end{cases} \tag{2.4}$$

Where $P^u_\alpha = M_{1w}...M_{q\alpha}$ is a path connecting camera C_α to the world frame represented by camera C_1.

2.2.3 Evolution

Once we have the hypotheses and the payoff function, we can perform the evolutionary process needed to select all consistent triangulations.

Let $x = (x_1, ..., x_n)^T$ be a discrete probability distribution over the available strategies H, which are our hypotheses. This vector represents the population vector and lies in the n-dimensional standard simplex $\Delta^n = \{x \in \mathbb{R}^n : x_i \geq 0 forall i = 1...n, \sum_{i=1}^n x_i = 1\}$.

The support of a population $x \in \Delta^n$ is denoted by $\sigma(x)$ and is defined as the set of all elements of x with non zero probability: $\sigma(x) = \{i \in [1, n] : x_i > 0\}$.

In order to find a set of mutually coherent hypotheses we are interested in finding configurations of the population such that the average payoff is maximized.
The total payoff obtained by hypothesis i within a given population x is

$$(\Pi x)_i = \sum_{j=1}^{n} \pi_{ij} x_j \tag{2.5}$$

So if we consider all the hypotheses, the weighted average payoff over all the hypotheses x is exactly

$$f(x) = x^T \Pi x \qquad st \quad x \in \Delta^n \tag{2.6}$$

Which is an objective function from a standard quadratic programme; it is not immediate to find the global optimum of $f(x)$ in the standard simplex but local maxima can be obtained using a class of evolutionary dynamics called *Payoff Monotonic Dynamics*. A common evolutionary process starts by setting an initial population x near the barycentre of the simplex and then continue by evolving its values through the discrete-time replicator dynamic:

$$x_i(t+1) = x_i(t) \frac{(\Pi x(t))_i}{x(t)^T \Pi x(t)} \tag{2.7}$$

Where x_i is the $i-th$ element of the population, Π the payoff matrix and $x_i(t)$ denotes x_i at time t. The sequence of population vectors that we obtain is guaranteed to stay and to evolve inside the standard simplex and it is ensured to converge to an equilibrium where the support does not include strategies with mutual payoff equal to zero. This means that the constraints introduced in equation 2.4 are actually enforced.
Usually, the equation 2.7 is iterated over the initial uniform population, then it can be stopped when the differences between population values at time t and at time $t-1$ are smaller than a given threshold.
At the end of the process, when the equilibrium is reached, the density in the final population vector can be used to assess the degree of participation of each hypothesis in the support.

This approach has shown to be very successful in addressing a wide range of problems, including feature based matching [3], medical images segmentation [28], rigid [5] and non-rigid [49] 3D object recognition.

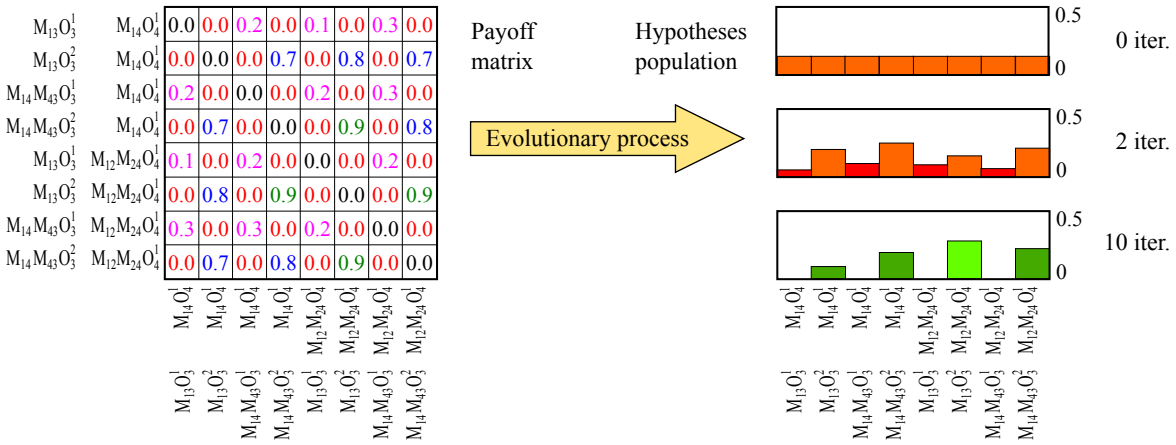

Figure 2.2: Example of the selection process applied to a small instance of problem shown in figure 2.1. In this very simplified example the payoff matrix is not perfectly accurate but the evolution process is computed accurately. After few iterations the distribution converges to a local optimum.

In figure 2.2 a complete, small case is illustrated assuming the network topology and observations shown in figure 2.1.
We assume that all pairwise extrinsic calibrations have been performed with good accuracy, except for M_{14} which is characterized by a large error for some reasons that we ignore. Observations O_3^2 and O_4^1 are correctly labelled, but still subject to an unknown measurement error. Observation O_3^1 is an outlier and results from a wrong labelling.
The set of cameras is $C = \{C_3, C_4\}$ and a total of four paths, two for each camera. The set of paths is $M = \{M_{13}, M_{14}, M_{43}, M_{14}, M_{12}, M_{24}\}$. Consequently we have a total of eight hypotheses, which are shown in figure.
Payoff matrix is shown with different colours in its entries: red is used to highlight entries which have been set to zero due to the constraints from expression 2.4. For instance, hypotheses $(M_{13}, O_3^1, M_{14}, O_4^1)$ and $(M_{13}, O_3^2, M_{14}, O_4^1)$ are not compatible since they include different observations from the same camera. Another couple of non compatible hypotheses is $(M_{13}, O_3^1, M_{14}, O_4^1)$ and $(M_{13}, O_3^1, M_{12}, O_4^1)$, which have zero consensus since they are connecting camera C_4 to the common world frame through different paths.
Hypotheses which received low mutual consistency due to geometrical inconsistencies are coloured in purple, and they have low payoff: basically they include pairs affected by the wrong labelling of O_3^1. Other pairwise payoffs are assigned according to coherence between triangulations. Blue entries are slightly lower that green ones due to the fact that they include the transformation M_{14}, which is less accurate than the others.

In the right side of figure 2.2 the actual evolution of the population through replicator dynamics is shown. The process starts from a uniform distribution, where each hypothesis has the same possibility to dominate. After just two iterations of equation 2.7, we can notice that hypotheses that come from outliers start to decrease, due to they low average payoff value. After ten iterations only feasible hypotheses survived.
Note that paths including the inaccurate transformation M_{14} are less represented in final population. The final distribution can be used to produce a weighted average of the final 3D points, rather than just selecting the point with the higher weight.

2.2.4 Shortcomings of OPS method

The principal shortcomings of OPS are due to the definition of the hypotheses set H. These hypotheses contain by construction a triangulated point computed from the two points, which value is used to compute the payoff value. Since the method works by validating triangulations, it needs to know in advance which projections to triangulate, so a candidate labelling method must be provided as input.
We know from previous section that such labelling can be obtained in several ways, but this strict requirement is a limitation for the utility of the proposed method. With this restriction we can only select an optimal path for each triangulation. We also know that the correspondences could be very bad in some cases, thus poor feature associations could lead to an increment of false positive observations and to an overall reduction of the algorithm performance.

Another main drawback of Optimal Path Selection is related to its complexity in terms of both memory and computational requirements. Each hypothesis is based on a combination of two observations through two different paths, even if we had a perfect labelling process in which every association is exact, the number of hypothesis grows with the square of the possible paths. This is due again to the "double" structure of such hypotheses: they must be created considering each possible couple of paths for each couple of observations with the same label.
The side of matrix Π is equal to the size of the hypotheses set $|H| \simeq |P|^2$, so each iteration of equation 2.7 is potentially $O(|P|^4)$. This is also true for the size in memory of the hypotheses and of the matrix Π, albeit it is not mandatory to actually store the entries of Π in memory since they can be computed directly during the evolutionary process, but this will increase the overall number of computations to perform.

These problems could alter the performance of the method, moreover we have a significant limitation on the number of hypotheses we can generate: if we consider a very wide camera network and many labelled couples with some uncertainty, we can have a significant loss both in computational time and in quality of the result. Finally, assuming a huge dataset with longer paths, the computation becomes infeasible.
In the following section we introduce a simple enhancement which can avoid these problems in both terms of capabilities and scalability. The new formulation can provide equally good results as in OPS method but at the same time it excludes the above drawbacks.

2.3 Joint Path Selection and Feature Labelling

The key idea is that the preliminary triangulation in each hypothesis can be totally avoided by adopting a simpler and less strict hypotheses set [47].
The proposed enhancement consists to base the selection process on a new set H where each hypothesis is defined as the ray resulting from the combination of a possible path P and an observation O.
Since this new kind of hypothesis corresponds to a line in space rather to a triangulated point, the defined payoff functions 2.3 and 2.4 need to be rearranged according to the new hypothesis formulation: in fact we can not compute the Euclidean distance between triangulated points since we have only one line in each hypothesis.

Before introducing the new payoff function, we have to redefine the set of hypotheses $H = \{H_1, H_2, ..., H_k\}$. The new formulation of an hypothesis is a couple (P_i, O_i), where P_i is a path which include, as usual, a sequence of rigid transformations from camera C_i reference frame to camera C_1 reference frame, and O_i is simply an observation from camera i. Note that we half the elements of the hypothesis and, for each camera we are able to build a distinct hypothesis for each observed point. In this way we are not forced to use labelling information simply because it is not required at this step.

We now introduce a reduced formulation of previous payoff function, substituting 2.3 with a function which only accounts for the skewness of the rays.

$$\pi'(H_i, H_j) = \pi'((P^u_\alpha, O^a_\alpha), (P^v_\beta, O^b_\beta)) = e^{-\frac{1}{2}(\frac{s(H_i,H_j)^2}{\sigma_s^2})} \tag{2.8}$$

Where $s(H_i, H_j)$ is the skewness value between the two lines induced by the two observations from the two hypotheses.

Of course this new measure is weaker than the first, since it is much easier for two rays to exhibit low skewness by chance, in fact this happens for all the rays lying in the same epipolar plane for each pair of cameras. However, if we have a high number of cameras, for a large enough population of candidate rays, the probability of exhibiting a low skewness by chance among all the possible pairs of cameras is much lower.

In this new formulation we reduce the size of the set H to the square root of previous section. No early triangulation is performed between pairs of observations so we can include more than one material point at the same time in our candidates and let the evolutionary process selects the clusters of rays belonging to the same bundle.
Note that the method which detected the observations could offer some a priori information about the likelihood of two observations to be related. For example if feature descriptors are used as observations we have a similarity measure between them which can be used to assess the similarity of two descriptors. Another example is when we have a tracking application and features could be associated through rules derived from camera motion. If such information can be provided from outside, a further improvement can be introduced. Without loss of generality, a compatibility function C can be defined, with $C(H_i, H_j) : H \times H \rightarrow \{0, 1\}$.This function indicates the feasibility of the correspondence according to the a-priori information. Consequently, the value of this function is zero when the two hypotheses are not compatible, while it is one when they could be the same material point.

Exploiting this compatibility knowledge, the complete payoff function can be defined as

$$\pi(H_i, H_j) = \pi((P^u_\alpha, O^a_\alpha), (P^v_\beta, O^b_\beta)) =$$

$$= \begin{cases} 0 & \text{if } C(H_i, H_j) = 0 \text{ or } \alpha = \beta \\ \pi'(H_i, H_j) & \text{otherwise} \end{cases} \tag{2.9}$$

Where, in addiction to compatibility function C, two observations are discarded if they came from the same camera.
Note that if an a-priori information about feature similarity is not provided, the compatibility function can simply be set to return always one: in this

way all possible couple of hypotheses are tested with each other.

The first iteration of the evolutionary process based on replicator dynamics with the payoff function 2.9 will yield a single material point and the same label can be applied to all supporting rays (e.g. rays which will be in the support of the final population).
Once the first group is obtained, two actions must be undertaken:

- the final point must be triangulated according to the rays that are still present in the final population. This can be done in several ways. In the simpler approach the two rays with the higher density are triangulated computing the closest 3D point to both rays. A more sensible and generic approach is to find the point that minimizes the squared distance from all the rays, weighted according to the population density of each ray. In this latter approach all final rays are considered but if some of them do not contribute in a significant way in the population (for example, if an outlier is included), they will not alter the final point computation.

- the non extinct rays must be removed from the hypotheses set H since they have already been assigned to a material point.

Once H has been reduced, additional iterations of the evolutionary process can be performed, until all the observation has been labelled or a satisfying number of material points has been reconstructed.

Chapter 3

Experimental Evaluation

In order to test both Optimal Path Selection (OPS) [46] and Joint Observation and Projection Selection (JOPS) [47] a set of synthetic camera networks have been designed in such a way that they resemble typical real-world camera network topologies. Testing in synthetic case allows to exclude all unpredictable error sources and properly analyse the behaviour of the methods with respect to erroneous observations and inaccurate extrinsic or intrinsic calibrations.

Three different network topologies were generated: they are depicted in figure 3.1.

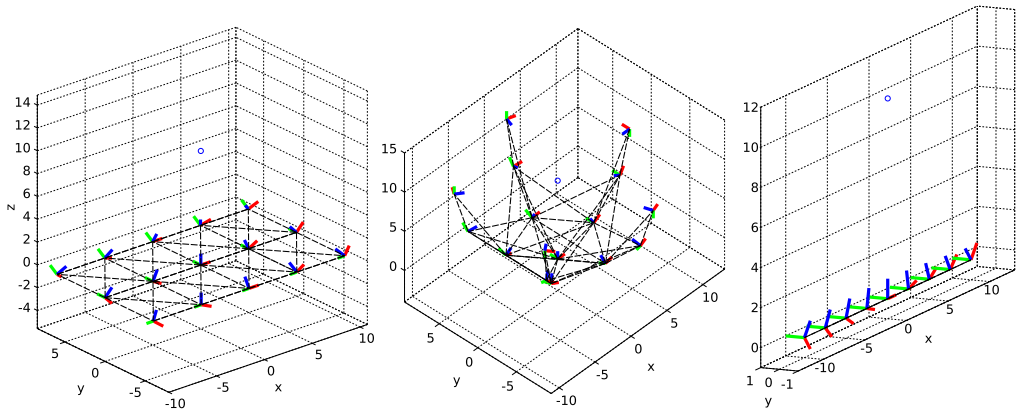

Figure 3.1: The three topologies used for synthetic experiments. From left to right: grid, hemisphere and line.

The first topology on the left is grid: it contains 15 cameras in an area of 20 $\times$ 12, centred at the origin of world coordinate system and lying onto the xy-plane. All cameras are rotated so that their z-axis points toward the

network centre $\mathbf{c} = (0, 0, 10)^T$. For this graph we generated the exact relative motion between all couples of cameras whose distance is less than 10 steps, for a total of 88 graph edges.
The second topology (figure 3.1, centre) is called hemisphere and it includes 16 cameras disposed in the surface of a semi-sphere with centre $\mathbf{c}$ and radius equal to 10. Origins of camera reference frames are placed with an uniformly distributed angular azimuth and elevation and all point to the sphere centre. A total of 90 edges describe the relative motions of camera pairs that are less than 10 steps apart.
Last generated topology is line (figure 3.1, right); it is composed by 9 cameras lying on the x-axis, uniformly spaced around the origin and oriented toward $\mathbf{c}$. A set of 30 edges links together the adjacent cameras.
For all three topologies intrinsic parameters were set with unitary focal length and principal point lying at the origin. Camera C_1 was placed so that its reference frame corresponds to world reference frame.

From this ground truth, many different perturbated instances of the various topologies were generated. This displacement was computed generating a normally distributed additive angular error applied to the rotation matrix of rigid motions associated to each edge. The distributions have zero mean and standard deviation σ_r. Various values of σ_r will be considered in order to simulate very small errors or more significant ones.

3.1 Optimal Path Selection

The first experiment was designed in order to analyse the sensitivity of the proposed method with respect to the parameters σ_p and σ_s, in payoff function defined in equation 2.3.
A random 3D point $\mathbf{P}$ was generated as ground truth in the neighbourhood of $\mathbf{c} = (0, 0, 10)$ and the spatial distance between $\mathbf{P}$ and the reconstructed 3D point was computed for any possible couple of parameters σ_p and σ_s in a certain range of values. The topology grid was used and an angular error with $\sigma_r = 9.1 * 10^{-3}$ was applied.
Figure 3.2 shows the results for $\sigma_s, \sigma_p \in [0.01, 0.2]$, respectively in x and y axis. As expected, there exists a large area around $\sigma_p = 0.08$ and $\sigma_s = 0.12$ where the interplay between the two parameters leads to satisfactory results. We have also an evidence that the skewness value, even if it is not a quantity directly related to the reconstruction error, can help the effectiveness of the payoff function in detecting the best hypotheses.

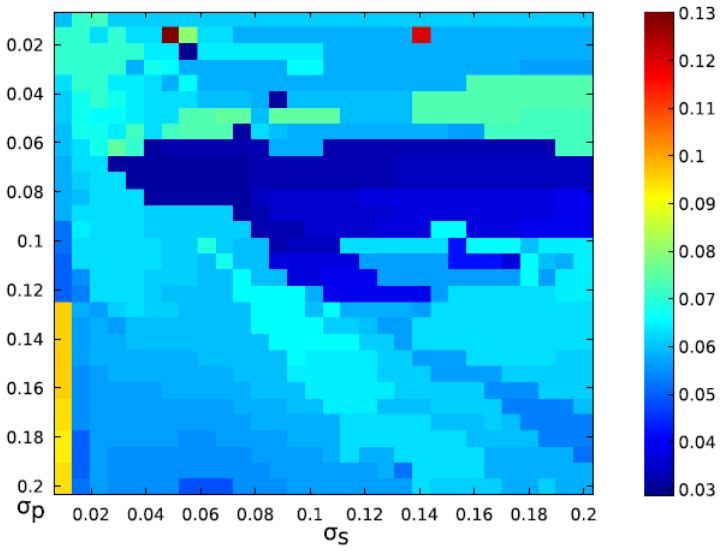

Figure 3.2: Sensitivity analysis of payoff parameters σ_s and σ_p with respect to the reconstruction error. We can notice an area (dark blue in the colour scale) where the algorithm performed better.

In the second experiment we compared the reconstruction accuracy of our approach against "dual-quaternion" [56] and "SBA" (Sparse Bundle Adjustment) [36] methods. The first exploits properties of dual quaternions to diffuse the camera network error and it creates a new set of coherent motions to a common reference frame. The latter is commonly used in structure from motion applications to simultaneously optimize extrinsic camera parameters and the reconstructed 3D points given their observations in each image plan.

Reconstruction error is evaluated for all grid, hemisphere and line topologies, varying the network graph angular error σ_r and observation error σ_o. The observation error was applied to all the observation coordinates through an additive zero mean Gaussian error with variance σ_o^2.
All the tests were performed by reconstructing 10 points generated in a random way around $\mathbf{c} = (0, 0, 10)$ as ground truth. Results for grid topology are shown in figure 3.3.
Since dual-quaternion works only on the orientations of each camera in the network, for this method we considered as measure of quality the best triangulation in terms of reprojection error between all the graph paths with less than three vertices each. Reprojection error is computed by projecting the reconstruction of the 3D point back to image plans of all cameras and

measuring the distance from the original observation.
In SBA approach, all the structure points were optimized at the same time. We can observe that SBA, even if it has the advantage of recomputing the camera poses while triangulating, mostly suffers from the observation errors specially in the case of outliers. On contrary, our proposed approach dynamically discards incoherent observations from the initial population, producing structures that are less noisy and more reliable. A similar behaviour can be observed for dual-quaternions method varying the standard deviation σ_r of graph edge error. Since it can only diffuse rotation errors without discarding the problematic edges, it suffers from large relative motion displacements that may happen during the graph calibration phase.

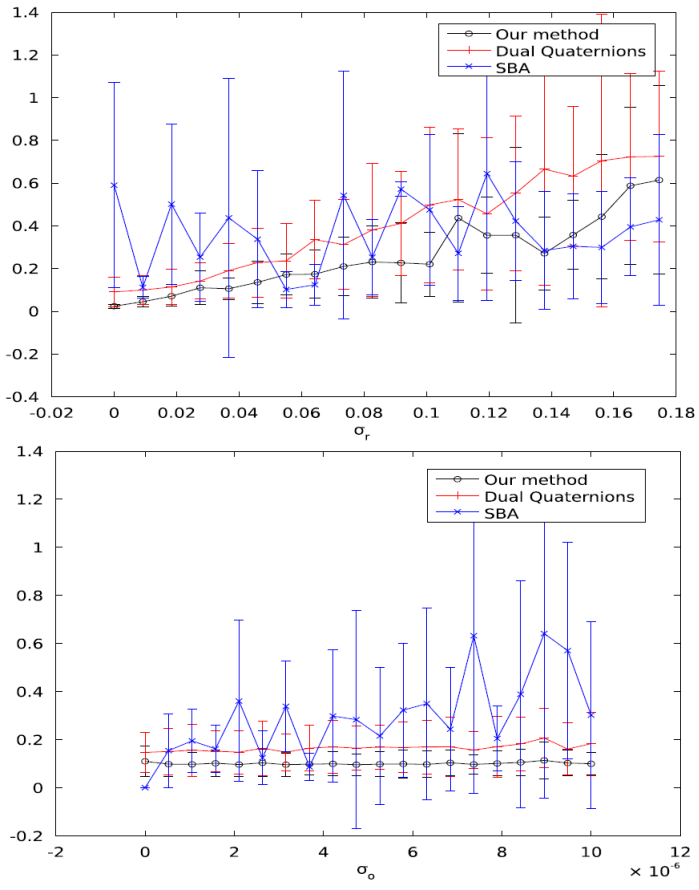

Figure 3.3: Performances of OPS, SBA and dual-quaternion methods for the grid topology. Triangulation error is computed while perturbing the graph edges (top) and varying the observation errors (bottom).

Overall, Optimal Path Selection can deliver the best of the two proposed methods, being either able smooth the errors by averaging the triangulation among many cameras while still being very effective when selecting a small set of mutual reliable paths and observations.

As last experiment, we tested the behaviour of the algorithm in the case of observations outliers. We computed the number of false positives and false negatives.
A false positive is a wrong observation still included in the final population and consequently, involved in the computation of the final 3D point. A false negative is an observation which originally was part of the correct observations but it has been wrongly discarded so that it do not participate in the point reconstruction. We can have false positive or false negatives for many reasons which can be related to camera orientation errors or we can have a bad initial labelling of the observed image features.
The number of false positives and false negatives are computed from the final population for many different triangulations attempts, varying the observations outlier distribution in the grid network topology.

To this end, we generated each time as ground truth exactly one inlier observation with a random uniform uncertainty of $\sigma_o = 10^{-3}$, and one outlier which was displaced from the ground truth observation by a factor of $K\sigma_o$. The more the parameter K is increased, the more the distance on the image plan between inlier and outlier observations increases. A false positive was counted every time an observation from the outlier point was included in the final population, and a false negative was counted when an observation from the correct point was excluded from the final population.
In figure 3.4 the results are plotted for false positives (left) and false negatives (right). We can immediately notice that the relative number of both false positives and negatives decrease proportionally with K. With $K = 3$, the number of wrongly selected observations is almost zero. In a real-case scenario, we expect two possible cases: if we have wrong observations very close to the correct one, then they should not influence the final triangulation. If they are more than 3 times the standard deviation far from the correct point, they will be discarded from the final population.

In the next section performances of the improved method are presented in order to prove that even using only the skewness value in the computation of payoff, we are still able to correctly reconstruct one or more 3D points. We will also introduce the feature selection process and set up a new test configuration.

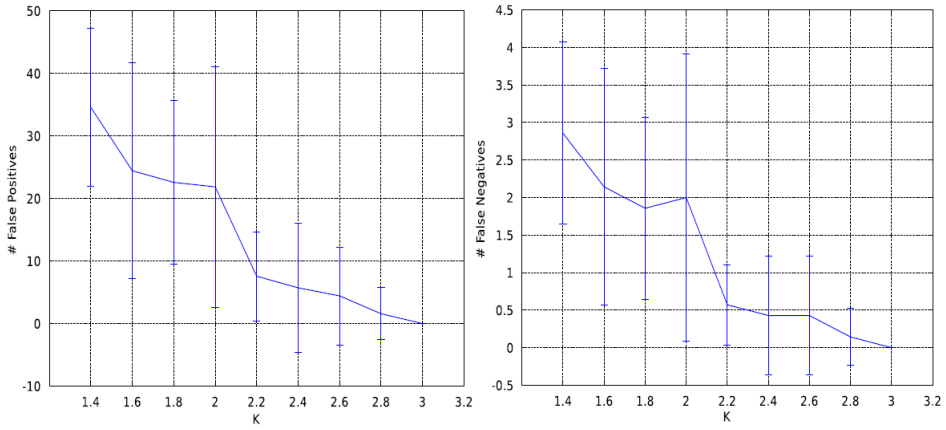

Figure 3.4: Number of false positives (left) and false negatives (right) while changing the parameter K. As K increases, the outlier point was shifted further from the correct observation. We can notice an inverse linear relation between K and the both the number of false positives and negatives.

3.2 Joint Optimal Path Selection and Feature Labelling

Beside the reduced complexity, there is no technical or theoretical reason that guarantees the proposed enhanced to perform better or at least on par than the previous one.
The only sensible way to assess the properties and the advantages of this improvement is to perform an extensive and complete set of comparisons over a various set of experimental conditions.

The final goal of each reconstruction technique is to accurately recover the geometry of the observed scene regardless of the error sources. For this reason in the next synthetic experiments we adopted as the measure of the result quality the RMS (Root Mean Square) or Quadratic Mean. It is defined as the square root of the arithmetic mean of the squares of a set of numbers, in our case the errors for each reconstructed point.
Assume we have a set of ground truth points $\{x_1, x_2, ..., x_n\}$ and the set of their estimated reconstruction is $\{\tilde{x}_1, \tilde{x}_2, ..., \tilde{x}_n\}$. Their RMS is computed as follows

$$RMS = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(x_i - \tilde{x}_i)^2} \tag{3.1}$$

In all the following results, RMS error was computed between the generated ground truth model and the reconstructed set of points.
Note that some methods we compare against might change the reference system due to modifications of camera poses. In order to be invariant with respect to these effects, RMS has always been computed after a best-fit alignment between the obtained 3D points and and the original model. This quality measure is called P_{rms}.

In order to be able to study different aspects of the proposed method, we designed a general synthetic experimental framework which allows to control a wide range of different hindrances that could affect the process in a real-world scenario.
In a particular way it is critical the ability of controlling the density of generated material points, their visibility ratio from one or more different cameras, the different noise sources, the amount and quality of the outliers. In fact, if we are working in a real application we can have the cameras spread around a wide area, so we need to model the portion of the scene that each camera can detect. Moreover, if we have dynamical or static objects to track from the environment, the presence of outlier is a common situation and we would like to filter them out from the final solution.

A graphical overview of this simulated scenario is presented in figure 3.5, where the different control parameters are also included.
The parameters that we modified during the experimental evaluation are the following:

- N_p (Number of points): the number of material points that have been generated in the model which are observed by the cameras. Like in the previous evaluation setup, such points are generated uniformly in a cube box of side 10 around the centre $\mathbf{c} = (0, 0, 10)$;

- V_r (Visibility ratio): the ratio of material points that can be observed from each camera. This parameter is added in order to model complex networks, especially when we are dealing with motion scenarios, when only a portion of the points are captured by each camera;

- σ_r (Orientation rotational error): this corresponds to the value from the previous section, it is the standard deviation of the rotational noise added to each camera in the network. We add only rotational noise

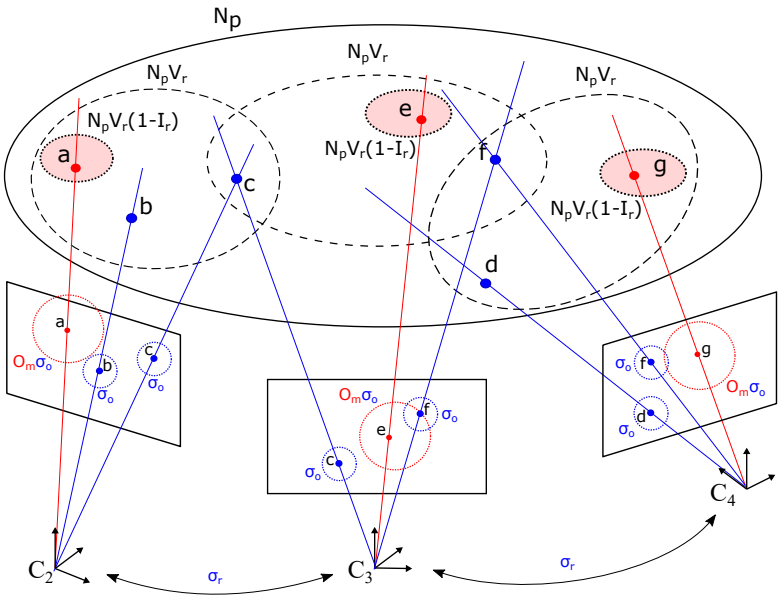

Figure 3.5: Synthetic setup with the six principal factors that are included.

because we want to allow a single measure of noise while still addressing the most influential factor;

- σ_o (Observation positional error): as before, it is the standard deviation of the positional noise added to each observation in the image plans;
- I_r (Inlier ratio): the ratio of observed observations that can be classified as inliers in each camera. Consequently, $1 - I_r$ can be thought as the outlier ratio for each camera;
- O_m (Outlier multiplier): the multiplier to be applied to σ_o in order to amplify the positional error of outliers.

According to this values, each camera can observe exactly $N_p V_r$ material points from the scene, which are chosen at random among the initial N_p. Among these visible points, the projections of exactly $N_p V_r I_r$ are subject to an additive Gaussian positional error with zero mean and standard deviation σ_o. The remaining points are considered outliers and their observations are displaced by a larger positional error modelled as a zero-mean Gaussian with a standard deviation equal to $\sigma_o O_m$.
In figure 3.5 these positional errors are well represented with red and blue dotted circles: red circles express more uncertainty in the position of the observation since they represent outliers.

We will work with the already described topology grid, hemisphere and line. All presented evaluation have been computed by creating a ground truth of points using the specified parameters and then averaging the P_{rms} obtained over the three topologies.
As with the previous method, we first performed an analysis of the sensitivity of the method to its payoff parameter, then we compared the new technique with the previous following the same baseline.

Unlike Optimal Path Selection, the Joint Observation and Projection Selection method depends on a single payoff parameter σ_s. In order to test the method with respect to this parameter, the average P_{rms} was computed for different values of σ_s with the following fixed experimental conditions: $N_p = 10$, $V_r = 0.9$, $\sigma_r = 0.018$, $\sigma_o = 7 * 10^{-3}$, $I_r = 0.9$ and $O_m = 10$.

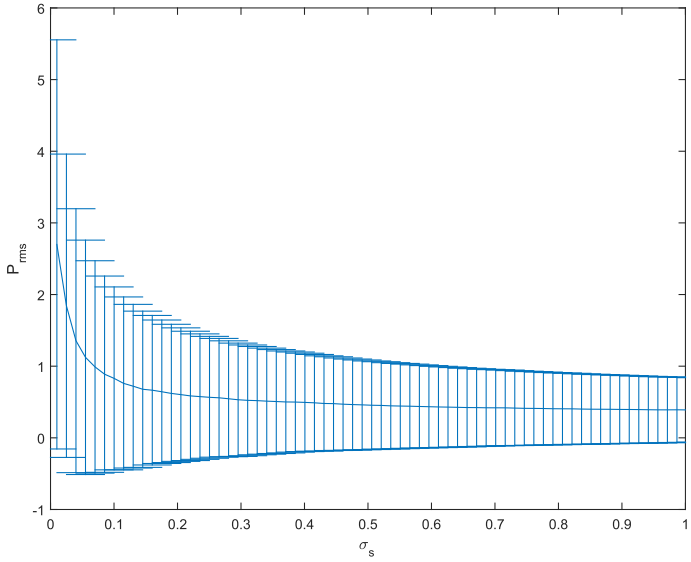

Figure 3.6: Average P_{rms} changing the payoff parameter σ_s: when σ_s is above a certain threshold, the reconstruction is not really influenced.

Results are shown in figure 3.6, where σ_s ranges from a value very close to zero to one. We can notice that JOPS is not very sensitive to σ_s as in the previous analysis where we had also σ_p. In fact, as long as σ_s is large enough to avoid a too strict inlier selection we had almost the same result.
This behaviour is still similar to the one exhibit by OPS but as it depends only by a single parameter we had a better tuning process.

We can also notice that, while increasing the value of σ_s above about 0.5 has a limited effect on P_{rms}, the shape of the final population could be very different for different values of the parameter. In figure 3.7 the (ordered) final population distribution is shown for two different values of σ_s. In the left plot the value of σ_s is 0.1 and we see that only few candidates are selected by the evolutionary process: note that they all have high probabilities associated. In the right plot we have $\sigma_s = 1$: the final population is more numerous with respect to the previous case, moreover all candidates have almost the same probability.
Smaller values of σ_s are more restrictive, in fact they lead to a tighter selection with less remaining hypotheses. On the other hand, higher values of σ_s are too permissive, allowing many hypotheses to enter in the final population with equal probability.

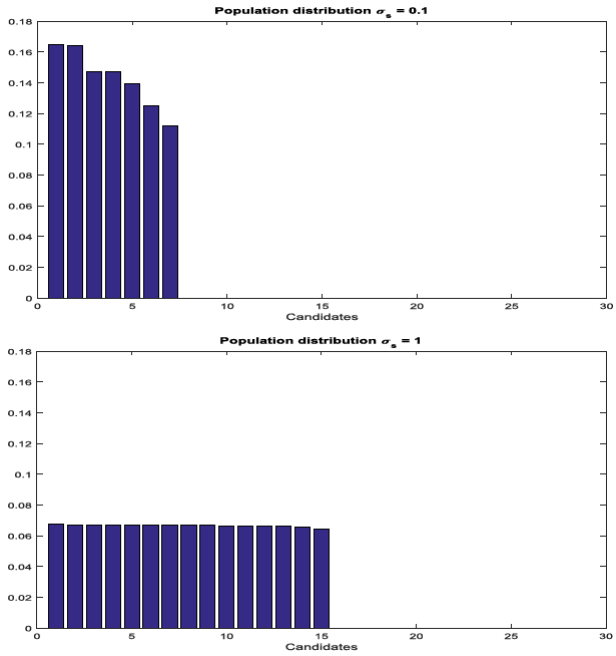

Figure 3.7: Distributions of sorted final population values when $\sigma_s = 0.1$ (top) and $\sigma_s = 1$ (bottom)

When we have few hypotheses as result, we could have a less stable triangulation because of the reduced number of samples contributing in the average. Moreover, we could have a duplicate detection of the same physical points because the remaining hypotheses could lead to the same result in a new iteration of the replicator dynamics.

For these reasons it is very important to tune properly the parameter σ_s, since we want a good number of relevant rays in order to perform a good reconstruction of the 3D point, but at the same time we do not want to include hypothesis which came from outliers or other points which could alter the goodness of the result.

In the evaluation part, results of JOPS have been compared with the previously introduced OPS, SBA (Bundle Adjustment) and dual quaternion averaging.
Since other methods can not perform feature matching but they need to start from some given correspondences, in order to compare JOPS with them for now we have to reduce the scope of our method.
We thus define a labelling l according to the actually observed points and a compatibility function from this labelling is used in JOPS. The compatibility function C is defined as follows

$$C(H_i, H_j) = \begin{cases} 1 & \text{if } l(H_i) = l(H_j) \\ 0 & \text{if } l(H_i) \neq l(H_j) \end{cases} \tag{3.2}$$

In this way we have full compatibility if the given labels are the same, and no compatibility if labels are different. Regarding the other methods, they have been feeded directly with the correct matches.

The sensitivities to different error sources have been studied separately, by exploring the obtained P_{rms} varying only one error source and keeping all the other parameters fixed as in the previous experiments.

Figures 3.8 and 3.9 are similar to figure 3.3. Several points have been generated for each network topology then the average of the three configurations was computed. The already proposed methods are now compared with JOPS as we increase levels of noise for the observation and for camera orientations. Values from JOPS are on par with OPS and are better than both SBA and dual quaternion methods.

Then we tested the performances of all four methods varying the Visibility ratio V_r. The tests was done generating 10 points at random in all different topologies. Visibility ratio V_r was increased starting from 0.3 (each camera can detect only 3 randomly selected points) to 1 (all cameras are able to see all 10 points). Results are shown in figure 3.10. We can see that JOPS performances are aligned with OPS, performing better than the

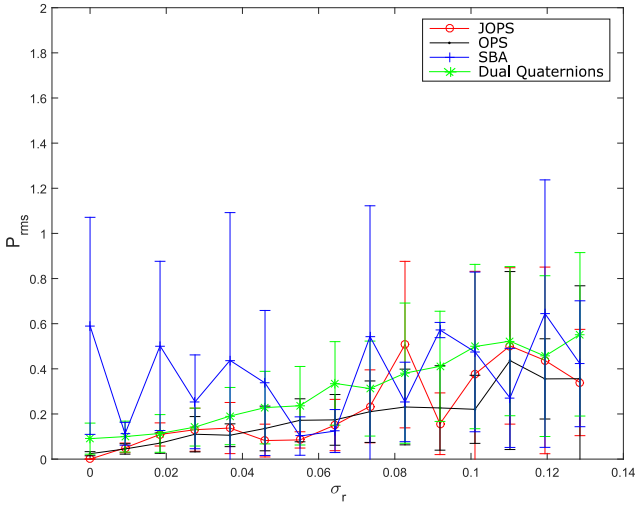

Figure 3.8: RMS values for all methods with respect to σ_r

other two methods. Note that SBA fails for values of V_r which are below 0.6: this because this method requires several points to be observed in order to compute its best approximation. This makes our method more effective in large and sparse camera networks, where we could have a significant number of occlusions and we need to reconstruct the scene.

In figure 3.11 the reconstruction precision is computed as the inlier ratio I_r is decremented.
We first set a low value for the outlier multiplier O_m: as before, the performances of JOPS are very similar to performances of OPS and both are much better than the other two methods. As the inlier ratio is decremented, SBA method tend to perform very badly, because it is not resilient to outliers. Also the standard deviations of both SBA and dual quaternions are very large.
In the other plot we set a higher value for O_m: we can see that JOPS and OPS perform as usual except for very extreme values, dual quaternions offers poor results and SBA does not even give a result because of the many outliers.

Figures 3.10 and 3.11 highlight the shortcomings of SBA and dual quaternions with respect to the proposed OPS and JOPS. SBA method is unable to reconstruct the original points where not enough correct observations are available, while dual quaternions is unable to avoid outliers since it just deals with camera orientation without considering observations.

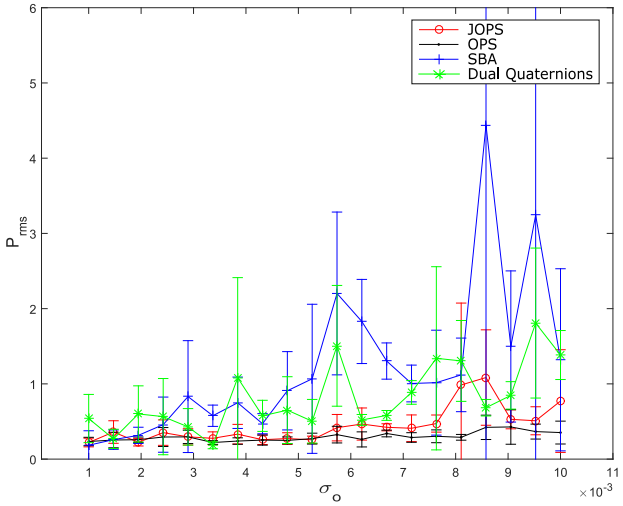

Figure 3.9: RMS values for all methods with respect to σ_o

The final and most important test has been designed to simulate a more general scenario with about 200 cameras and a scene of about 1000 material points producing unlabelled observations.

We assumed that the observations are characterized by an hypothetical descriptor vector that can be used as an initial filter to detect unfeasible correspondences. For this hypothetical descriptor we want to simulate different levels of repeatability and distinctiveness.
In order to implement these features, a suitable compatibility function can be defined in the following way:

$$C(H_i, H_j) = \begin{cases} X_r \sim Ber(R_f) & \text{if } l(H_i) = l(H_j) \\ X_d \sim Ber(1 - D_f) & \text{if } l(H_i) \neq l(H_j). \end{cases} \tag{3.3}$$

Where $X \sim Ber(p)$ means that X is a binary random variable modelled as a Bernoulli distribution of parameter p, which corresponds to the probability that $X = 1$.
In the case that the labels given by l are equal, the compatibility will be described by a binary random variable X_r having value 1 with a probability R_f. On the other hand, if the given labels are not equal, compatibility will be described by the binary random variable X_d that is 0 with probability D_f

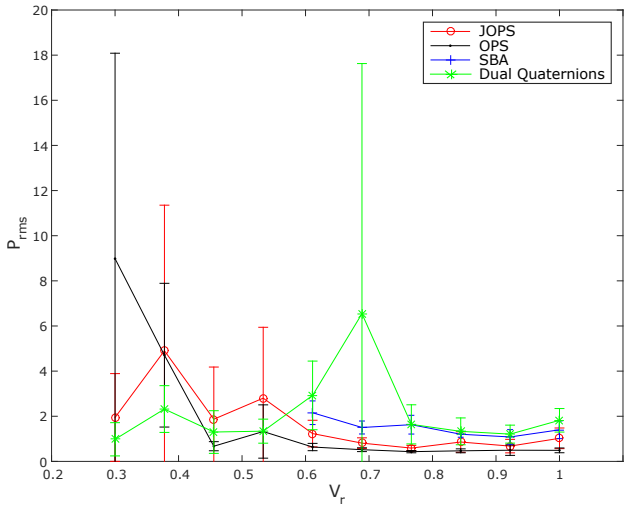

Figure 3.10: RMS values for all methods with respect to visibility ratio V_r

and 1 with probability $1 - D_f$.
R_f is the repeatability factor that models the probability that the descriptors computed over two observations of the same physical points are equally labelled. Thus, $1 - R_f$ expresses the probability to be unable to recognize a matching feature.
The value D_f stands for distinctiveness factor and represent the probability that the projections of two different material points are actually considered non corresponding.
According to this parametrization, the exact compatibility function corresponds to setting $R_f = 1$ and $D_f = 1$. By contrast, if we set both to zero, we will have a completely wrong descriptor.

Figure 3.12 shows the average P_{rms} obtained with JOPS method for different combinations of R_f and D_f values. We can observe that JOPS behaves very well also with relative low values of R_f and D_f, which rarely occurs if we choose a well designed descriptor.
Moreover, when the repeatability is high enough, we can notice that distinctiveness is less critical since, as long as we have correct correspondences in the set of hypotheses H, outliers given by low values of D_f can be easily filtered out by the evolutionary process because of the lack of geometrical consistency.

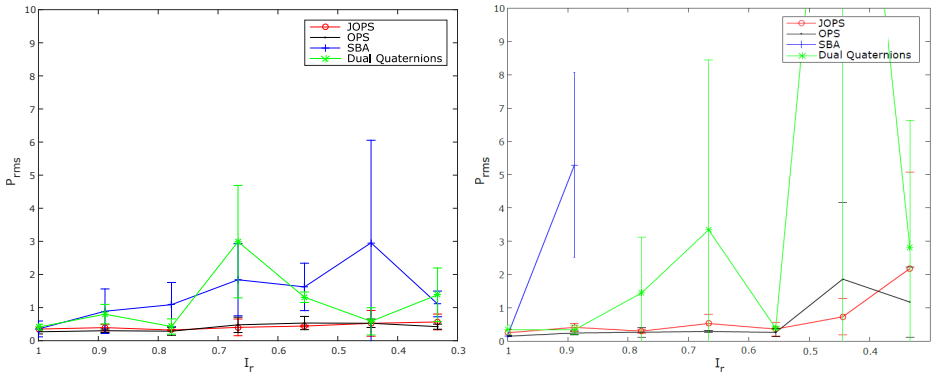

Figure 3.11: RMS values for all methods with respect to inlier ratio I_r. Different outlier multipliers was used: small O_m in left plot, high O_m in the right one.

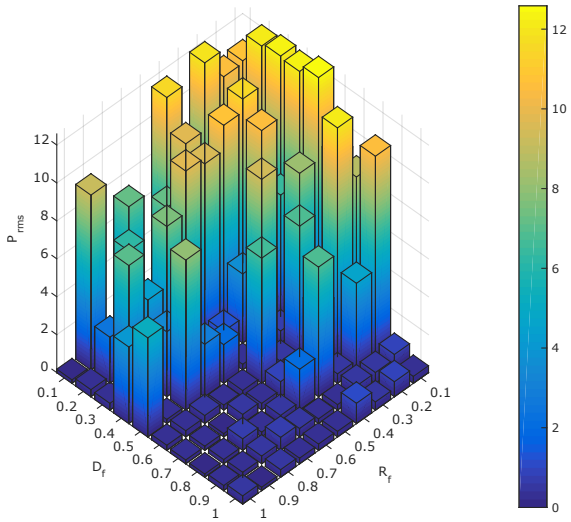

Figure 3.12: Performances of JOPS for different degrees of simulated feature descriptor repeatability and distinctiveness.

Chapter 4

Conclusions

In this thesis a dynamic path selection method has been introduced. This method could be used to perform robust 3D reconstructions when dealing with a camera network possibly suffering from inaccurate extrinsic calibrations. To this end, we presented two different approaches.

The first proposed method performs point-wise reconstruction by dynamically selecting the best possible set of camera poses and observations that maximize the consistency of each pairwise triangulation. Each candidate in this approach is composed by a couple of possible paths from two cameras connecting them to the common world reference frame and by a couple of observations that are triangulated using such paths.
The ability to locally exclude part of the graph or observations from the final triangulation makes our method particularly effective in all the scenarios when either a precise calibration can not be provided or only a poor localization of targets is available.
Note that this approach is also reliable in the case of outliers, since they are discarded by the game-theoretical evolutionary process. Notice finally that the method also assumes to have an initial correspondences labelling available as a starting point, albeit it exhibits some degree of tolerance with respect to labelling errors.

The second approach is an improvement in terms of spatial and temporal computation. It offers the same advantages of the first proposed method but it is also able to perform the selection of correspondences between point projections without an initial labelling. At the same time it still selects the optimal camera orientations required to triangulate such observations.
This enhanced method can be adopted in more scenarios than the simple Optimal Path Selection, such as large camera networks, sequences of frames

or collection of images from the web.
In general, both approaches make no assumptions about the method used to obtain the initial pose estimation between the cameras, neither they make assumptions regarding the technique used to capture material point images. However, if some kind of descriptor or similarity function is available, the joint approach can easily exploit them in order to obtain a better reconstruction with the best precision available from the pose estimations.

The experimental evaluation of the method has shown that it is able to offer a performance quality comparable to similar approaches which are addressed to a narrower application range and have major shortcomings.
Specifically, the novel approach is very resistant to outliers and very bad edges. In fact, other methods just propose a kind of adjustment of the overall parameters and observations perform worse simply because they are forced to include the very bad calibrated edge or the correspondence errors; while in our methods if there is an hypothesis which is in stronger disagree with all the others, it will be discarded in a very fast way and excluded from the final result.

Finally, when adopted to solve the more general problems of simultaneous observations and paths selection, the JOPS method exhibited a strong resilience even when dealing with feature descriptors characterized by low distinctiveness and repeatability.

Bibliography

[1] Sameer Agarwal, Noah Snavely, Ian Simon, Steven M Seitz, and Richard Szeliski. Building rome in a day. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 72–79. IEEE, 2009.

[2] Alexandre Alahi, Raphael Ortiz, and Pierre Vandergheynst. Freak: Fast retina keypoint. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 510–517. Ieee, 2012.

[3] Andrea Albarelli, Samuel Rota Bulo, Andrea Torsello, and Marcello Pelillo. Matching as a non-cooperative game. In *Computer Vision, 2009 IEEE 12th International Conference on*, pages 1319–1326. IEEE, 2009.

[4] Andrea Albarelli, Luca Cosmo, Filippo Bergamasco, and Andrea Torsello. High-coverage 3d scanning through online structured light calibration. In *Pattern Recognition (ICPR), 2014 22nd International Conference on*, pages 4080–4085. IEEE, 2014.

[5] Andrea Albarelli, Emanuele Rodola, Filippo Bergamasco, and Andrea Torsello. A non-cooperative game for 3d object recognition in cluttered scenes. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on*, pages 252–259. IEEE, 2011.

[6] BS Alsadik, M Gerke, and G Vosselman. Optimal camera network design for 3d modeling of cultural heritage. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 3, 2012.

[7] Dana H Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern recognition*, 13(2):111–122, 1981.

[8] Joao Barreto and Kostas Daniilidis. Wide area multiple camera calibration and estimation of radial distortion. In *Proceedings of the 5th Workshop on Omnidirectional Vision, Camera Networks and Non-Classical Cameras, Prague, Czech Republic*, 2004.

[9] Herbert Bay, Andreas Ess, Tinne Tuytelaars, and Luc Van Gool. Speeded-up robust features (surf). *Computer vision and image understanding*, 110(3):346–359, 2008.

[10] Matthew Brown and David G Lowe. Recognising panoramas. In *ICCV*, volume 3, page 1218, 2003.

[11] Matthew Brown and David G Lowe. Unsupervised 3d object recognition and reconstruction in unordered datasets. In *3-D Digital Imaging and Modeling, 2005. 3DIM 2005. Fifth International Conference on*, pages 56–63. IEEE, 2005.

[12] Peter J Burt and Edward H Adelson. The laplacian pyramid as a compact image code. *Communications, IEEE Transactions on*, 31(4):532–540, 1983.

[13] John Canny. A computational approach to edge detection. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (6):679–698, 1986.

[14] Jiawen Chen, Dennis Bautembach, and Shahram Izadi. Scalable real-time volumetric surface reconstruction. *ACM Transactions on Graphics (TOG)*, 32(4):113, 2013.

[15] Yang Chen and Gérard Medioni. Object modelling by registration of multiple range images. *Image and vision computing*, 10(3):145–155, 1992.

[16] Brian Curless and Marc Levoy. A volumetric method for building complex models from range images. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, pages 303–312. ACM, 1996.

[17] Richard O Duda and Peter E Hart. Use of the hough transformation to detect lines and curves in pictures. *Communications of the ACM*, 15(1):11–15, 1972.

[18] O Elkhalili, OM Schrey, W Ulfig, W Brockherde, BJ Hosticka, P Mengel, and L Listl. A 64ã 8 pixel 3-d cmos time of flight image sensor for car safety applications. In *Solid-State Circuits Conference, 2006. ESSCIRC 2006. Proceedings of the 32nd European*, pages 568–571. IEEE, 2006.

[19] Pedro F Felzenszwalb and Daniel P Huttenlocher. Efficient graph-based image segmentation. *International Journal of Computer Vision*, 59(2):167–181, 2004.

[20] Yasutaka Furukawa and Jean Ponce. Accurate, dense, and robust multiview stereopsis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(8):1362–1376, 2010.

[21] S Burak Gokturk, Hakan Yalcin, and Cyrus Bamji. A time-of-flight depth sensor-system description, issues and solutions. In *Computer Vision and Pattern Recognition Workshop, 2004. CVPRW'04. Conference on*, pages 35–35. IEEE, 2004.

[22] Peter Hammerstein, Reinhard Selten, et al. Game theory and evolutionary biology. *Handbook of game theory with economic applications*, 2:929–993, 1994.

[23] Chris Harris and Mike Stephens. A combined corner and edge detector. In *Alvey vision conference*, volume 15, page 50. Citeseer, 1988.

[24] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[25] Janne Heikkila and Olli Silvén. A four-step camera calibration procedure with implicit image correction. In *Computer Vision and Pattern Recognition, 1997. Proceedings., 1997 IEEE Computer Society Conference on*, pages 1106–1112. IEEE, 1997.

[26] Christof Hoppe, Andreas Wendel, Stefanie Zollmann, Katrin Pirker, Arnold Irschara, Horst Bischof, and Stefan Kluckner. Photogrammetric camera network design for micro aerial vehicles. In *Computer vision winter workshop (CVWW)*, volume 8, pages 1–3, 2012.

[27] Eli Horn and Nahum Kiryati. Toward optimal structured light patterns. *Image and Vision Computing*, 17(2):87–97, 1999.

[28] Bulat Ibragimov, Bostjan Likar, Franjo Pernus, and Tomaz Vrtovec. A game-theoretic framework for landmark-based image segmentation. *Medical Imaging, IEEE Transactions on*, 31(9):1761–1776, 2012.

[29] Omar Javed, Khurram Shafique, and Mubarak Shah. A hierarchical approach to robust background subtraction using color and gradient information. In *Motion and Video Computing, 2002. Proceedings. Workshop on*, pages 22–27. IEEE, 2002.

[30] Pakorn KaewTraKulPong and Richard Bowden. An improved adaptive background mixture model for real-time tracking with shadow detection. In *Video-based surveillance systems*, pages 135–144. Springer, 2002.

[31] Sohaib Khan and Mubarak Shah. Consistent labeling of tracked objects in multiple cameras with overlapping fields of view. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(10):1355–1360, 2003.

[32] Dieter Koller, Gudrun Klinker, Eric Rose, David Breen, Ross Whitaker, and Mihran Tuceryan. Real-time vision-based camera tracking for augmented reality applications. In *Proceedings of the ACM symposium on Virtual reality software and technology*, pages 87–94. ACM, 1997.

[33] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2548–2555. IEEE, 2011.

[34] Marc Levoy, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, Jeremy Ginsberg, et al. The digital michelangelo project: 3d scanning of large statues. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 131–144. ACM Press/Addison-Wesley Publishing Co., 2000.

[35] Ze-Nian Li, Mark S Drew, and Jiangchuan Liu. *Fundamentals of multimedia*. Springer, 2004.

[36] Manolis IA Lourakis and Antonis A Argyros. Sba: A software package for generic sparse bundle adjustment. *ACM Transactions on Mathematical Software (TOMS)*, 36(1):2, 2009.

[37] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[38] David Marr and Ellen Hildreth. Theory of edge detection. *Proceedings of the Royal Society of London B: Biological Sciences*, 207(1167):187–217, 1980.

[39] K. Mikolajczyk and C. Schmid. A performance evaluation of local descriptors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(10):1615–1630, Oct 2005.

[40] Greg Mori, Xiaofeng Ren, Alexei A Efros, and Jitendra Malik. Recovering human body configurations: Combining segmentation and recognition. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–326. IEEE, 2004.

[41] AA Morye, C Ding, Bo Song, A Roy-Chowdhury, and Jay A Farrell. Optimized imaging and target tracking within a distributed camera network. In *American Control Conference (ACC), 2011*, pages 474–480. IEEE, 2011.

[42] John Nash. Non-cooperative games. *Annals of mathematics*, pages 286–295, 1951.

[43] Martin J Osborne. *An introduction to game theory*, volume 3. Oxford University Press New York, 2004.

[44] Federico Pedersini, Augusto Sarti, and Stefano Tubaro. Accurate feature detection and matching for the tracking of calibration parameters in multi-camera acquisition systems. In *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*, volume 2, pages 598–602. IEEE, 1998.

[45] Pietro Perona and Jitendra Malik. Scale-space and edge detection using anisotropic diffusion. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 12(7):629–639, 1990.

[46] Mara Pistellato, Filippo Bergamasco, Andrea Albarelli, and Andrea Torsello. Dynamic optimal path selection for 3d triangulation with multiple cameras. In *Image Analysis and Processing—ICIAP 2015*, pages 468–479. Springer, 2015.

[47] Mara Pistellato, Filippo Bergamasco, Andrea Albarelli, and Andrea Torsello. Robust joint selection of camera orientations and feature projections over multiple views. In *23rd International Conference on Pattern Recognition (ICPR 2016), Cancun, Mexico, December 4-8, 2016 (submitted)*, 2016.

[48] Sricharan Ramagiri, Rahul Kavi, and Vinod Kulathumani. Real-time multi-view human action recognition using a wireless camera network. In *Distributed Smart Cameras (ICDSC), 2011 Fifth ACM/IEEE International Conference on*, pages 1–6. IEEE, 2011.

[49] Emanuele Rodola, Alex M Bronstein, Andrea Albarelli, Filippo Bergamasco, and Andrea Torsello. A game-theoretic approach to deformable shape matching. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 182–189. IEEE, 2012.

[50] Jianbo Shi and Jitendra Malik. Normalized cuts and image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):888–905, 2000.

[51] Martin Shubik. *Game theory in the social sciences: Concepts and solutions*, volume 155. JSTOR, 1982.

[52] Chris Stauffer and W Eric L Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2. IEEE, 1999.

[53] Murali Subbarao and Gopal Surya. Depth from defocus: a spatial domain approach. *International Journal of Computer Vision*, 13(3):271–294, 1994.

[54] Gopal Surya and Murali Subbarao. Depth from defocus by changing camera aperture: A spatial domain approach. In *Computer Vision and Pattern Recognition, 1993. Proceedings CVPR'93., 1993 IEEE Computer Society Conference on*, pages 61–67. IEEE, 1993.

[55] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Science & Business Media, 2010.

[56] Andrea Torsello, Emanuele Rodola, and Andrea Albarelli. Multiview registration via graph diffusion of dual quaternions. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 2441–2448. IEEE, 2011.

[57] Emanuele Trucco and Alessandro Verri. *Introductory techniques for 3-D computer vision*, volume 201. Prentice Hall Englewood Cliffs, 1998.

[58] Pieter G Van Dokkum. Cosmic-ray rejection by laplacian edge detection. *Publications of the Astronomical Society of the Pacific*, 113(789):1420, 2001.

[59] Xiaogang Wang. Intelligent multi-camera video surveillance: A review. *Pattern recognition letters*, 34(1):3–19, 2013.

[60] Jörgen W Weibull. *Evolutionary game theory*. MIT press, 1997.

[61] Oliver Wulf and Bernardo Wagner. Fast 3d scanning methods for laser measurement systems. In *International conference on control systems and computer science (CSCS14)*, pages 2–5. Citeseer, 2003.

[62] Anthony Yezzi Jr, Satyanad Kichenassamy, Arun Kumar, Peter Olver, and Allen Tannenbaum. A geometric snake model for segmentation of medical imagery. *Medical Imaging, IEEE Transactions on*, 16(2):199–209, 1997.

[63] Zhengyou Zhang. A flexible new technique for camera calibration. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(11):1330–1334, 2000.

[64] Song Chun Zhu and Alan Yuille. Region competition: Unifying snakes, region growing, and bayes/mdl for multiband image segmentation. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 18(9):884–900, 1996.