Network camera tracking, a possible containment of a pandemic

Group: 5, components: Davide Allegro, Davide Barbieri, Davide Ghion

Automation Engineering, Networked Control for Multi-Agent Systems

University of Padua, Padua, Italy

Abstract

The Covid-19 pandemic that has hit the whole world this year has made anyone think about how to prevent and at least partially avoid its spread. Especially many closed environments such as fairs, airports and shopping centers and many others, have been forced to remain closed for a long time due to the inability to control any gatherings and to avoid entry to people with fever.

The solution adopted to prevent entry to people with fever, was to engage a worker at each entrance with the aim of measuring the temperature of each customer. Another possibility is represented by using a camera network to solve and automate this process without engaging human collaborators. A good starting point to apply this kind of solution is a common place shared by a large number of people, indeed where there is a continue flow, *e.g.* an airport.

In this relation it is showed a simplified simulation of these real scenarios, described above, indeed it is considered a camera network that is installed in a map (arena) and the aim of this network is to detect and track the sick people (targets) in order to help those who supervise to avoid contagion.

Furthermore, once explained the way in which it is possible to do target tracking, it is focused on how it is possible to prevent possible errors related to the algorithm, the cameras and management of outliers.

I. Introduction

Nowadays, utility of video surveillance becomes a common fact in many fields. Among these surveillance systems, multiples camera are used to improve coverage and accuracy in the surveyed area. Due to the huge amount of data generated by these video surveillance systems, the need for automatic analysis techniques that may be used to exploit these data becomes more indispensable to study the behaviour of people without needing many human operators to analyze the captured videos.
A possible implementation is proposed in this relation, as matter of fact it is studied how a camera network can be used in order to track people with Covid symptoms, in particular measure the temperature of people and track who has it higher than $T_{sick} = 37.5$ °C.
The main advantage could be the reduction of personnel involved in tracking and limiting the spread of the infection in large enclosed space such as an airport or a shopping center, etc. . It is taken into account these big environments, because the access control is a complex problem and mistakes can compromise the number of infected people, for example there are various accesses to an airport, people could enter from different entrances and can come from a plane which stop over or from service doors for all staff. Until now, all airports have a vast camera network but it was not necessary to take the temperature of the people. The idea is to improve this architectures implementing an empowered camera that is composed by a thermal camera and depth camera, as reported in [1], in order to know both the temperature and the distance of the target.
Various methods had been proposed to achieve an accurate tracking in the most challenging conditions as occlusions and lighting variations. These methods are addressed in two main research lines: the centralized and the distributed tracking approaches.

Definition 1. *In a centralized system, each camera propagates his measurements to a central node and the latter merge all information received and take a decision.*

Definition 2. *In a distributed system, each camera can make decision about its measure and the information received from his neighbors. The decision aim to solve the problem of consensus.*

In this relation, the solution proposed uses a centralized algorithm because it is supposed to filter the acquired video sharing to the centralized controller only few data. Moreover in the real case, like the Venice airport the system is centralized, then this encouraged us to follow this way. It is also true that the distributed approach is more flexible and clearly more efficient, hence this study can be used as starting point to make comparison with smarter solutions.
As said above it is assumed that each node can synthesize the input data sharing to the central unit only

filtered data like the temperature of a person and his distance taken from the point of view.
In order to model the system, it is necessary to identify the state of such model, *i.e.* the position of a target. It is considered a random walk to express the possible movements of a person in an enclosed space. Then, given the distances measured by the cameras that are able to see the target it is adopted the Kalman Filter (KF) in order to estimate the state, namely the position.
For linear dynamic systems, the KF [2], which provides an optimal state estimation, is usually derived in the minimum-mean-square-error (MMSE) sense with the assumption of Gaussian uncertainties. However, as in our case, most systems in real world applications are nonlinear. Indeed, we know from the cameras the euclidean distances, of course this is not a linear problem. Then it is needed the Extended Kalman Filter (EKF) [3], [4].
Another important consideration is related to the possible problems of vision of the cameras, due to the breakage of some component or the aging that imply an inaccurate measure. Moreover there are many reasons that imply a miss detection of a target, for example some reflection, low light or some implementation problem due to the algorithm adopted. Then to overcome these problems it is applied a calibration step of our camera network in order to identify the cameras which in the most cases give us a noisy information that alter the estimate. In this way, it is possible to decrease the computational burden, removing the camera superfluous. Furthermore, it is considered an estimation of the quality of the vision and the status of each camera, in order to weight our state estimation giving more importance to the most reliable cameras.
Very briefly, the following points will be addressed in the next sections:

- Description of the models of the system, in the Sec. III;
- Illustration of the algorithm that solve the problems presented in the Sec. IV;
- Introduction of the multi-tracking problem and the recognition of a miss detection in the Sec. V;
- How we built the simulations, our comments on all results obtained and the conclusion in the Sec. VII-VIII;

II. STATE OF ART

Since the 1960s, video-surveillance systems have evolved in a parallel line to their automation grade and three generations can be clearly differentiated, as stated by Räty [5]: (1st) (1960–1980) Analog CCTVs and low level of automation. (2nd) (1980–2000) Digital CCTVs and computer vision processing. (3rd) (2000 – nowadays) Semi-automated video-surveillance systems.
The third generation of video-surveillance systems has achieved a certain grade of automation which allows to detect some suspicious human behaviors and to give off the corresponding alarms. In most

cases, these systems follow a similar pattern to define their actuation methodology based on several sequential steps, which mainly are the foreground objects detection, tracking and behavioral analysis, as reported in [6].

Regardless of all the advances which has involved the third generation of video-surveillance, there are some persistent high-level problems which block a higher grade of automation: the low cooperation between security systems, the extremely high valued assets insufficiently protected by obsolete technology or the excessive dependence on the intensive human concentration to detect and assess threats. Furthermore, there are other low-level problems which are not efficiently solved: the tracking in low-quality videos, occlusions between objects, algorithms executed in real-time, etc.

Nowadays, the video images captured from cameras strategically located are the principal element in any surveillance system. An automatic video-surveillance is mandatory in order to control a wide area. Moreover, cameras could acquire a lot of data leading to an overload of the network and then to delays not admissible in a real time tracking. For this reason, in this report, the focus is on a specific case where only few information are take into account and transmitted to the centralized processing unit. In order to have a clear point of view of the considered situation, the sensors (cameras) considered are smart devices, then all the capabilities are fully described by the standard *IEEE 1451*. Just to understand the argument, a smart device sends to a central processing unit the measures and to avoid high overhead the latter lead the reading of by certain events. This kind of communication management supposed to reduce the traffic load and the bandwidth consumption.

Furthermore the camera block is composed by a depth camera and a thermal camera, as matter of fact a camera capable of obtaining both information does not exist. There are several depth cameras available, in this relation it is considered the Intel RealSense Depth Camera D435. On the other side, the thermal cameras are already used to tracking, it is considered the FLIR Elara FR-345-EST, this camera is capable of capturing video in complete darkness, bright sunlight, and through smoke, dust, or light fog.

III. System Model

This architecture can be used in many different scenarios like in a airport or in a shopping center, we suppose to implement it in a wide area with many cameras and zones where one or more cameras can see. First of all, in our simulation we develop a model which is able to describe the random behaviour of a target in a wide space. Then we implement the Random Walk (RW) considering an unpredictable behaviour of the target, as matter of fact he could go anywhere he wants, in an airport he could go to the check-in, to a shop because he is an airport worker or he could exit and so on.

Definition 3. *A generic random walk is a mathematical object, known as a stochastic or random process, that describes a path that consists of a succession of random steps on some mathematical space such as the integers.*

Therefore, we consider a discrete-time nonlinear dynamic system with the hidden vector $\mathbf{x}_k \in \mathbb{R}^m$. At each time instant k, $\mathbf{x}_k$ is observed by J nodes in a network. This network is described by a time-invariant geometric graph $\mathcal{G} = \{\mathcal{J}, \mathcal{E}\}$, in which $\mathcal{J} = \{1, \dots, J\}$ and $\mathcal{E}$ denotes the set of nodes and edges, respectively. We assume that the network is connected which means each cameras is connected to the central processing unit, then our network has a star configuration. All these nodes are homogeneous in terms of processing and communication capability. The inter-node communication links are assumed to be ideal. The dynamic process and the local observation of each node $j \in \mathcal{J}$ can be described using the following state-space model:

$$\mathbf{x}_{k+1} = \mathbf{f}_k(\mathbf{x}_k) + \mathbf{G}_k\mathbf{w}_k \tag{1}$$

$$\mathbf{y}_{j,k} = \mathbf{h}_{j,k}(\mathbf{x}_k) + \mathbf{v}_{j,k} \tag{2}$$

where the state function $\mathbf{f}_k(\cdot) : \mathbb{R}^m \to \mathbb{R}^m$ and observation function $\mathbf{h}_{j,k}(\cdot) : \mathbb{R}^m \to \mathbb{R}^n$ are both differentiable and possibly nonlinear. Here, $\mathbf{G}_k \in \mathbb{R}^{m \times p}$ is a matrix. The vector $\mathbf{w}_k \in \mathbb{R}^p$ and $\mathbf{v}_{j,k} \in \mathbb{R}^n$ denote the process and observation noise, respectively. Both noise vectors are assumed to be zero mean Gaussian distributed with covariance

$$\mathbf{E}\left\{\begin{bmatrix}\mathbf{w}_k \\ \mathbf{v}_{j,k}\end{bmatrix}\begin{bmatrix}\mathbf{w}_l \\ \mathbf{v}_{i,l}\end{bmatrix}^T\right\} = \begin{bmatrix}\mathbf{Q}_k & \mathbf{0} \\ \mathbf{0} & \mathbf{R}_{j,k}\end{bmatrix}\delta_{ji}\delta_{kl}, \tag{3}$$

where δ_{ji} is Kronecker delta, i.e. $\delta_{ji} = 1$ only if $j = i$. The initial state is assumed to be $\mathbf{x}_0 = \mathcal{N}(\mathbf{m}_0, \mathbf{P}_0)$. To make full use of all information in the network, we define a collection of the observations of the entire network $\mathbf{y}_k = [\mathbf{y}_{1,k}^T, \dots, \mathbf{y}_{J,k}^T]^T \in \mathbb{R}^{nJ}$. Thus, the global nonlinear observation function w.r.t. the state $\mathbf{x}_k$ is defined by $\mathbf{h}_k(\mathbf{x}_k) = [\mathbf{h}_{1,k}^T(\mathbf{x}_k), \dots, \mathbf{h}_{J,k}^T(\mathbf{x}_k)]^T \in \mathbb{R}^{nJ}$. Similarly, the global observation noise is $\mathbf{v}_k = [\mathbf{v}_{1,k}^T, \dots, \mathbf{v}_{J,k}^T]^T \in \mathbb{R}^{nJ}$ with the block diagonal covariance matrix $\mathbf{R}_k = \text{blkdiag}[\mathbf{R}_{1,k}^T, \dots, \mathbf{R}_{J,k}^T]^T \in \mathbb{R}^{nJ \times nJ}$. Thus, the global observation model is

$$\mathbf{y}_k = \mathbf{h}_k(\mathbf{x}_k) + \mathbf{v}_k \tag{4}$$

by stacking the J nonlinear equations in (2).

The key objective is to infer the hidden state $\mathbf{x}_k$ at each time k based on a set of available observations $\mathbf{y}_{1:k} := \{\mathbf{y}_1, \dots, \mathbf{y}_k\}$. From a Bayesian viewpoint, at time k we want to recursively estimate the predictive distribution $p(\mathbf{x}_k|\mathbf{y}_{1:k-1})$ given observation up to $k-1$ and the filtering distribution $p(\mathbf{x}_k|\mathbf{y}_{1:k})$ given observation up to k, as a description of general Bayesian filtering (BF) [7].

IV. Centralized Extended Kalman Filter

The EKF provides us a tool for dealing with such nonlinear models in an efficient way. Since it is computationally cheaper than other nonlinear filtering methods such as point-mass filters and particle filters, the EKF has been used in various real-time applications like navigation systems.

Considering the global system model (1) and (4), the filtering densities are assumed to be approximated by Gaussian distribution [8], i.e. $p(\mathbf{x}_k|\mathbf{y}_{1:k}) \approx \mathcal{N}(\mathbf{x}_k|\hat{\mathbf{x}}_{k|k}, \mathbf{P}_{k|k})$ with filtered mean $\hat{\mathbf{x}}_{k|k}$ and covariance matrix $\mathbf{P}_{k|k}$. This evolution of Gaussian approximations is formed by linearizing the nonlinear function using the first-order Taylor series at current best state estimate, i.e., $\hat{\mathbf{x}}_{k|k-1} = \mathbb{E}[\mathbf{x}_k|\mathbf{y}_{1:k-1}]$ and $\hat{\mathbf{x}}_{k|k} = \mathbb{E}[\mathbf{x}_k|\mathbf{y}_{1:k}]$ in prediction and filtering step, respectively. Here, we use $\hat{\mathbf{x}}_{k|k-1}$ and $\mathbf{P}_{k|k-1}$ to express the mean and covariance matrix of Gaussian approximated predictive distribution, i.e., $p(\mathbf{x}_k|\mathbf{y}_{1:k-1}) \approx \mathcal{N}(\mathbf{x}_k|\hat{\mathbf{x}}_{k|k-1}, \mathbf{P}_{k|k-1})$. Thus, the linearized state-space model at time k is

$$\mathbf{x}_{k+1} \approx \mathbf{F}_k(\mathbf{x}_k) + \mathbf{G}_k\mathbf{w}_k \tag{5}$$

$$\bar{\mathbf{y}}_k \approx \mathbf{H}_k(\mathbf{x}_k) + \mathbf{v}_k \tag{6}$$

with Jacobian matrices $\mathbf{F}_k \in \mathrm{R}^{m\times m}$ and $\mathbf{H}_k \in \mathrm{R}^{nJ\times m}$:

$$\mathbf{F}_k = \left.\frac{\partial \mathbf{f}_k}{\partial \mathbf{x}_k}\right|_{\mathbf{x}_k=\hat{\mathbf{x}}_{k|k}}, \qquad \mathbf{H}_k = \left.\frac{\partial \mathbf{h}_k}{\partial \mathbf{x}_k}\right|_{\mathbf{x}_k=\hat{\mathbf{x}}_{k|k-1}} \tag{7}$$

as well as the deterministic input $\mathbf{u}_k$ and the reformulated observation vector $\bar{\mathbf{y}}_k$ defined by:

$$\mathbf{u}_k = \mathbf{f}_k(\hat{\mathbf{x}}_{k|k}) - \mathbf{F}_k\hat{\mathbf{x}}_{k|k} \tag{8}$$

$$\bar{\mathbf{y}}_k = \mathbf{y}_k - \mathbf{h}_k(\hat{\mathbf{x}}_{k|k-1}) + \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1} \tag{9}$$

Based on the linearized model (5) and (6), the filter is derived correspondingly [3]. With a proper initialization, the update equations of the EKF at each time k are as follows:

Prediction step

$$\hat{\mathbf{x}}_{k|k-1} = \mathbf{F}_{k-1}\hat{\mathbf{x}}_{k-1|k-1} + \mathbf{u}_k = \mathbf{f}_k(\hat{\mathbf{x}}_{k-1|k-1}) \tag{10}$$

$$\mathbf{P}_{k|k-1} = \mathbf{F}_{k-1}\mathbf{P}_{k-1|k-1}\mathbf{F}_{k-1}^T + \mathbf{G}_{k-1}\mathbf{Q}_{k-1}\mathbf{G}_{k-1}^T \tag{11}$$

Filtering step

$$\mathbf{P}_{k|k} = (\mathbf{P}_{k|k-1}^{-1} + \mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{H}_k)^{-1} \tag{12}$$

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{P}_{k|k}\mathbf{H}_k^T\mathbf{R}_k^{-1}(\bar{\mathbf{y}}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}) \tag{13}$$

with the Jacobian matrices (7) and modified observation (9).

In this relation we implement the Centralized EKF (CEKF), it imposes to compute a common estimation of the target performed by a centralized processing unit. The CEKF could cause to issue on the transmission of lot of data in a network, however in our real case we acquire only few data from a single camera block, then we evaluate it as a suitable solution for our specific simulation. Instead, the Distributed EKF (DEKF) is a research topic, it could be implemented in order to improve tracking, reduce the amount of data transmission and decrease connections, a single camera would be connected only with its neighbours. Moreover, we custom the EKF because at different time instants the system can acquire from a not fixed number of nodes. Even if we suppose that the target moves in random way, the matrices $\mathbf{H}_k$ and $\mathbf{R}_k$ depend by the number of active cameras, this simplification allow us to reduce the computational time. This is reasonable because only activated cameras acquire information that must be transmitted to the centralized processing unit, the other cameras do not have any information or they provide only noise.

V. Multi tracking

The objective of multi-target tracking is to simultaneously estimate the number of targets and their individual trajectories from sensor observations. Challenges in multi-target tracking include false alarms (clutter etc.), data association uncertainty (*i.e.* which target generated which measurement is unknown), and miss-detection. Moreover, we take into account for each node two different cameras that have to communicate with each other, then this situation could leads to other failures in the human tracking.
Another aspect that has to be examined is the human tracking across multiple cameras. When a person enters into the field of view (FOV) of a camera, human tracking within a camera is needed. However, when he/she leaves the FOV, the human information is no longer available, thus the limited FOV of a camera cannot meet the needs of wide-area human tracking. In order to cover wide areas, human tracking across multiple cameras has to be used, which helps to analyze global activities in the real world. Tracking human across multiple cameras includes two different scenarios:

Definition 4. *In the overlapping camera views there is a common FOV area between two cameras' views, where the target can be seen from both cameras.*

Definition 5. *In the non-overlapping camera views' scenario, there is not a common FOV area between two cameras' views, i.e., every camera's view is completely disjointed, and human cannot be seen in the so-called blind area.*

In our scenario we choose to employ the overlapping camera views because considering only few information provided by cameras, with non-overlapping camera views is not possible to estimate the real location of the target. As matter of fact, knowing only the distance (d) of the target from a single camera, the estimated point could be at any point on the circumference of radius $r = d$ and centered in the position of the camera. It is also true that to avoid this uncertainty the observation model can be modified by adding the angle in addition to the distance. Considering more information, as reported in [9], it is possible to estimate better the people flow and then implementing the tracking with a non-overlapping technique.

VI. Miss Detection

Recalling the measurement system, it is composed by two types of cameras that allow to detect and track a person with an internal temperature greater than a certain threshold T_{sick} and measure the distance. Here the basic idea that explain how is taken a measure:

1) The starting trigger is made by thermal camera that view a region of pixels that exceeds the threshold.
2) The depth camera receives the information by the thermal camera, it tries to detect the target and track his movements measuring the distance.

First of all it is needed to define what a miss detection is.

Definition 6. *Let's consider the camera's system described above, also consider that the thermal camera is completely reliable as a priori information, then when the depth camera can not detect the target this lead to a Miss Detection (MD).*

Clearly, it is supposed to minimize this kind of events in order to deal with reliable data, moreover it is needed a weight to associate to each camera to measure the reliability.

A. Empiric detection

A first solution can be made by the creation of a variable that controls if a camera have to be considered or not, indeed an ignored camera implies that it provides bad measurements.
To identify this variable it can be created a data-set of RW then compute the probability to have a good detection for each camera:

$$\beta_j = \frac{\#good\ detection}{\#total\ cases} \ \ ; \ \ \forall j = 1, ..., J. \tag{14}$$

Hence, all factors are stuck in a vector

$$B = [\beta_1, ..., \beta_J] \tag{15}$$

Moreover it is defined a quality threshold $T_q = 0.75$ that allow to discretize the vector B, in other words it is defined a probability of success $p = T_q$ and consider B as a vector of Bernoulli random variables *i.i.d* .

$$(B)_j = \begin{cases} 1 & , \beta_j \geqslant p \\ 0 & , \text{otherwise} \end{cases} \tag{16}$$

At this point the vector B can be viewed as a boolean vector that select certain cameras.

B. Weighted CEKF (WCEKF)

In this section, the WCEKF method is presented. In the previous section we introduced the possibility to remove from the beginning some cameras, that have provide bad measurements, from the trajectories estimation. By the way it is important to note that by discretizing the reliability factors β_j it is introduced an approximation error:

$$e_{d,j} = 1 - \beta_j \ \ ; \ \ \forall j = 1, ..., J \tag{17}$$

with $\beta_j \in [0, 1]$ computed in (14).

The more $e_{d,j}$ is large the more it is considered imprecise measurements. The WCEKF has a similar propagation and update model as the CEKF showed above. However it is an important difference: a multiplicative factor is added in the filtering step, the aim is to decrease the contribute of measurements with a large $e_{d,j}$, indeed to do this the innovation it is weighted by the coefficients proportional with β_j. The latter is applied for each measurement vector, $\bar{y}_k$ in order to create the correspond sensitivity matrix, H_k, before all the measurements are fused.

Then equation (13) become:

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + \mathbf{P}_{k|k}\mathbf{H}_k^T\mathbf{R}_k^{-1}\mathbf{B}_k(\bar{\mathbf{y}}_k - \mathbf{H}_k\hat{\mathbf{x}}_{k|k-1}) \tag{18}$$

where $\mathbf{B}_k = [\beta_1, ..., \beta_k]^T$ is the vector of the weights for the cameras that are seeing the target at the moment k.

To explain the reasoning, it is good to understand that to have a good estimation the innovation must be near zero, this means that the prediction $\hat{\mathbf{x}}_{k|k-1}$ is computed in a right way.

Hence, in the CEKF when the target position is shared between many cameras, the final estimation is computed by merging linearly all the contributes. It implies that, in case of bad measurements provided by one or multiple cameras, the $\hat{\mathbf{x}}_{k|k}$ value contains a large error contribution.

Although using $\mathbf{B}_k$ it can be possible to equalize the importance of the linearized measurement vector $\bar{\mathbf{y}}_k$, then the contribution of all the innovations that are too large, due to noise or other errors, are decreased and vice versa.

C. Camera health check

The definition of MD presented above is derived from the statistical point of view, however this not clarify what is a good detection or a bad one. In this section it is presented a possible self-diagnosis method, that can be installed in each camera, that tell us the health status of the camera, more precisely identify the reliability coefficient.
This method is represented by a *C++* program that use the *OpenCV* libraries, the basic idea is to take a photo of a known pattern, compare the differeces between a 100% working situation with the actual one. The comparison is performed by a feature matching algorithm, indeed the idea is to extract keypoints from the pair of images, compute descriptors and match them using a metric.

Definition 7. *A feature or keypoint is a meaningful, detectable part of the image, it can be a corner, blob or stable region. There exist many detectors, which must be stable, invariant to trasformations (rotation, traslation, etc...) and they should be insensitive to illumination change.*

Definition 8. *The descriptor provides a vector representation of a local region of an image, it is based on color, texture, orientation (computed by gradient operator). Moreover a descriptor have to be robust to noise and stable over viewing angle, illumination, blur, compression.*

Hence, the feature matching is performed by the *SIFT* algorithm, it is a widely used algorithm because it is very reliable keypoint detector and descriptor, furthermore the image content is mapped into local feature coordinates this means that it is invariant to: translation, rotation, scale and other transformations. Like said above, the first image considered it is the picture of the pattern taken with a perfectly working state, this act as best result that the camera aspire to reach. Then the second one is another picture of the pattern taken in the actual moment, this tell us about the current state of the camera.
In practice imagine that the camera can move around the *Y* axis, before the beginning of the algorithm it is request a fully rotation of the camera in order to bring optic's view to the roof where is placed the pattern, view Figure 1.

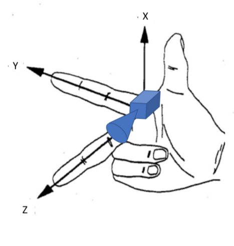

Figure 1. Right-handed 3D reference frames w.r.t. a camera

At the end of the *SIFT*, with the keypoints matched between the two images it is computed the index which exploit the health status of the camera, it is a simple percentage factor that count the good matches from the total cases.

Algorithm 1: Self-diagnosis algorithm

```
Result: Return a reliability coefficient $\beta$
initialization;
load the images: $I_{fixed}$ and $I_{actual}$;
detect and compute keypoints for $I_{fixed}$, $I_{actual}$;
perform SIFT feature matching;
while $\forall$ $keypoint$ do
    evaluate descriptors $descr_{I_{fixed}}$, $descr_{I_{actual}}$;
    if $d(descr_{I_{fixed}}, descr_{I_{actual}}) \in [0, 100]$ then
        mark keypoint as good match;
    else
        discrad keypoint;
    end
end
compute $\beta = \frac{\#\{good\ matches\}}{\#\{total\ matches\}}$
```

We have tried to stress a bit this kind of heuristic solution in order to simulate a realistic situation where, we know it is impossible to reproduce the original pattern having a perfect comparison. Here it is reported the interesting comparison made with the program.

1) Trasformation invariant test: To start the real comparison between the different situations, the first test's aim is to check if the algorithm is reliable and well calibrated. Indeed we created a fake image where the original pattern is scaled and rotated, in the figure above you can view that the recognition is acceptable with a good reliability factor.

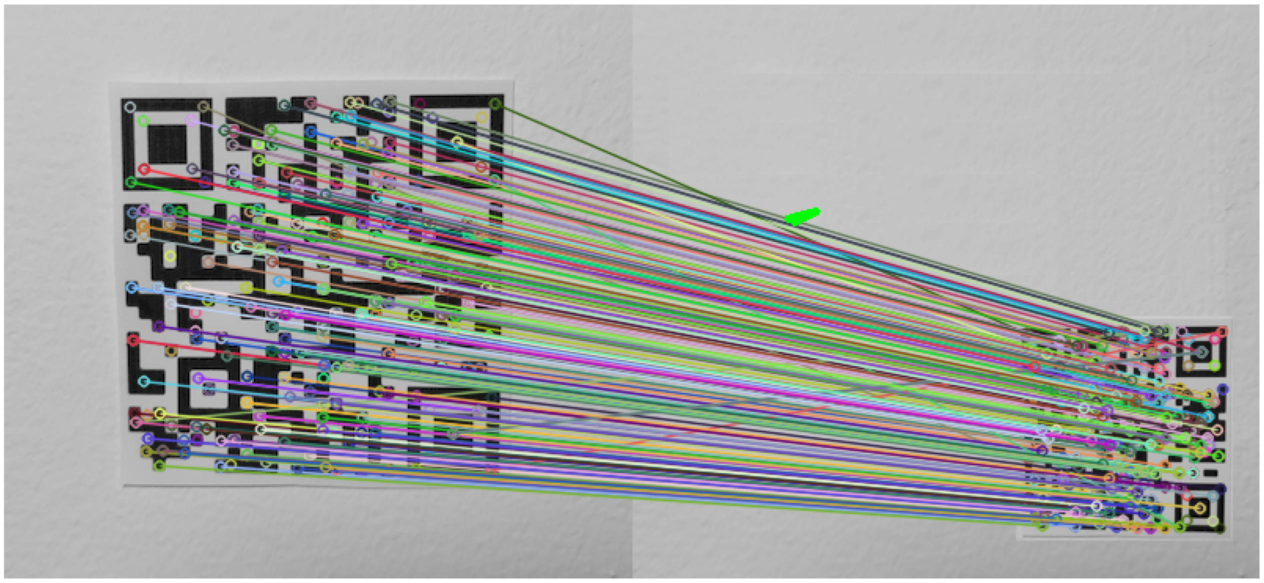

Figure 2. SIFT result with fake test, $\beta = 0.88$

2) Healthy camera test: Another tricky test that we made is to suppose that the motors that control the movement of the camera introduced some error, indeed the camera does not reach the desidered position, indeed there is a different illumination due to the different moment of the day, also in this case the algorithm find enough matches to consider the comparison acceptable, you can see the result in the figure below.

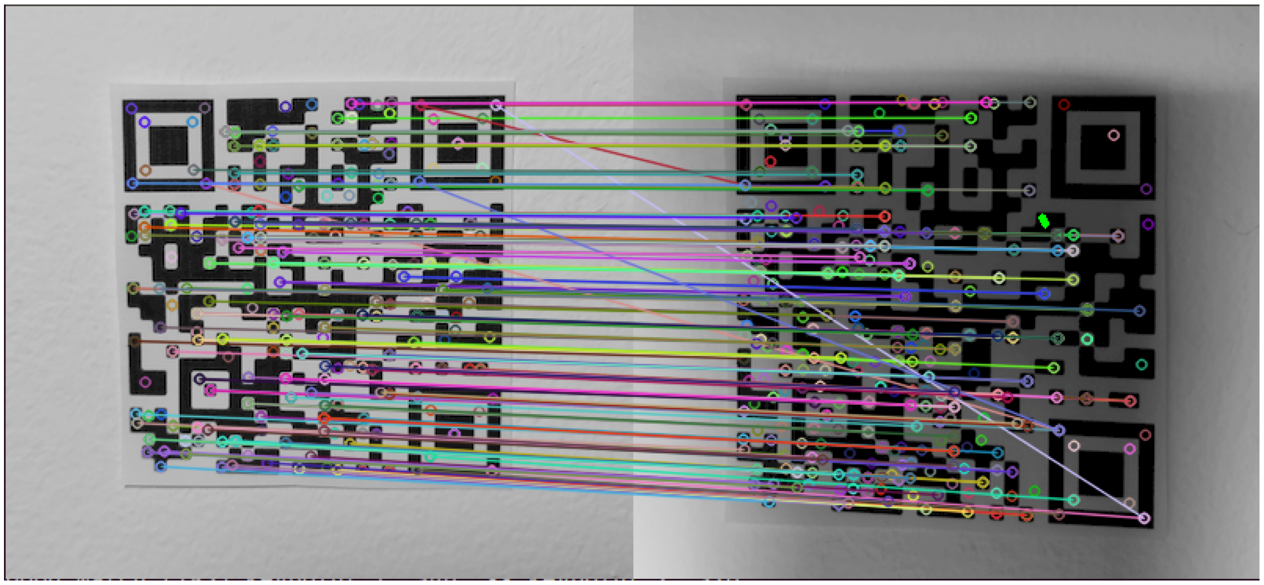

Figure 3. SIFT result with healthy camera, $\beta = 0.61$

3) Bad camera test: The final situation considered is when the camera optics need to be reconfigured, for example we supposed that the focal length is shorter than the necessary, this imply a bad focus of the pattern and indeed the camera will provide bad measurements.

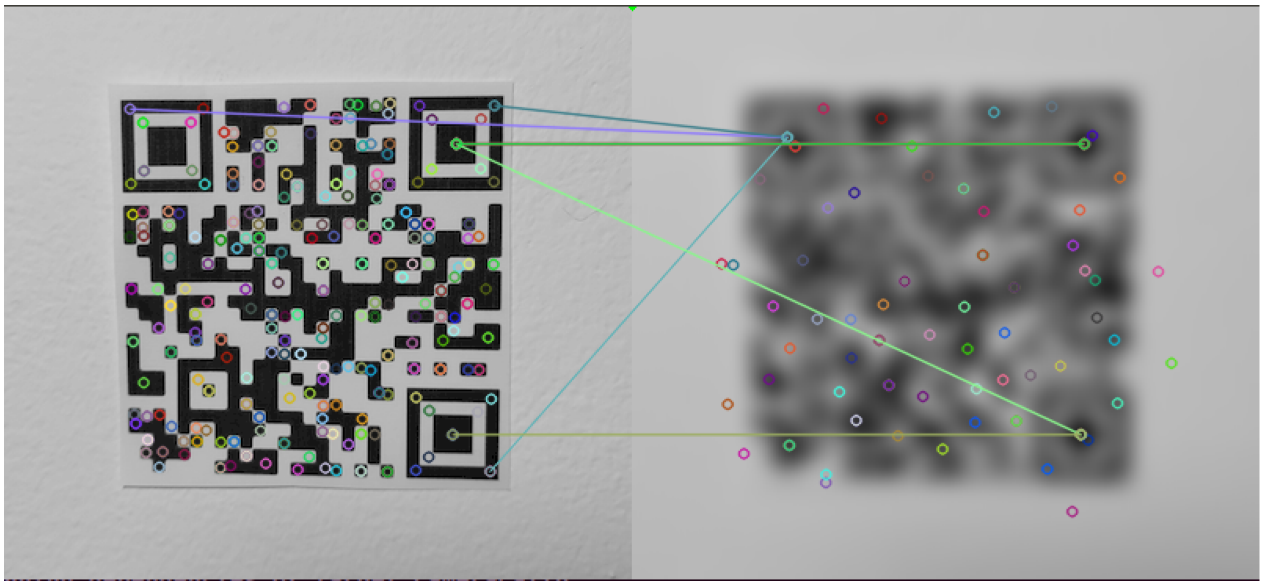

Figure 4. SIFT result with noisy camera, $\beta = 0.02$

VII. Simulation and Results

This section, first of all, is focused on the description of the environment for the simulation of our project. We analyze the structure of the camera network and the model adopted to simulate a real path of a person. Moreover we provide a comment of the results obtained adopting different typologies of the camera network.

A. Environment settings

As it is explained in the introduction, this project could be implemented in all possible place where it is very difficult to check and track the flow of people. Obviously in our simulation we could not represent an airport or an environment like a fair. For this reason in our simulation we consider a general arena with changeable sizes (rectangular or square environment). Moreover, in order to make the simulation more realistic, it is introduced an obstacle at the center of the virtual room which is inaccessible to people. The camera network setup follows a matrix distribution in order to cover homogeneously all the surface of the arena. Indeed the sensors on the contour of the matrix are placed equally spaced along the perimeter of the virtual room faced the obstacle and those inside the matrix are placed, faced towards the wall, along the contour of the obstacle. A possible version of the arena is shown in the following picture:

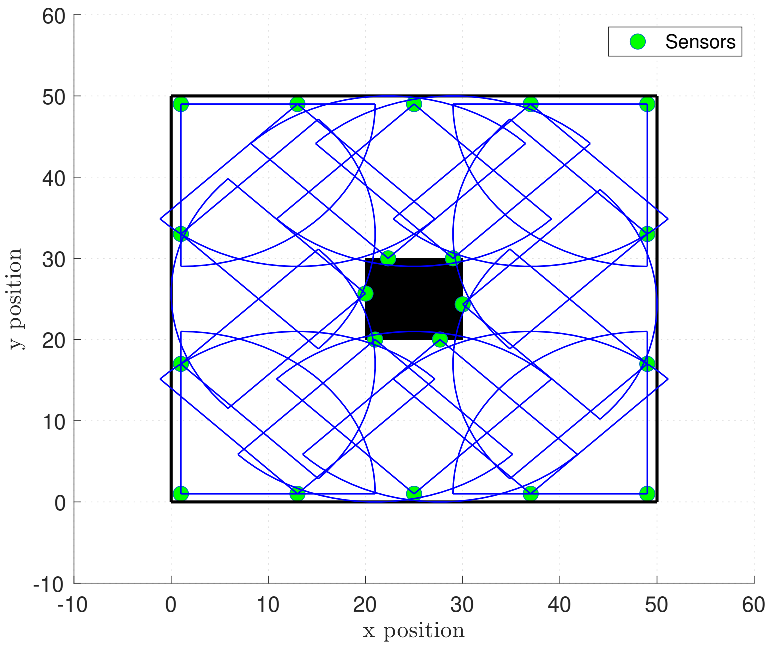

Figure 5. Example of camera network setup

In this arena 50×50, it is fixed 20 cameras (green markers). Fourteen of them are facing to the room center and the others are positioned on the obstacle (black square) in order to face towards the walls of the arena. It is assumed that each camera has limited vision, more precisely, the FOV can be described by a 90 ° wide cone with a radius $r = 20$, that can be set in a preliminary stage.

Obviously, as it is explained in the previous section V, the numbers of cameras and its arrangement are really important to get good results. It is showed in the next sections that implementing a not suitable camera network for a specific environment it is obtained bad results in term of unreliability accuracy. It is considered a bad setup for camera network when the target is viewed only by single camera for the most of the time. As it was explained in the section, Sec.V, when a target is seen only from one camera or even by no camera, the estimates could be not so accurate or even completely wrong. As matter of fact we want to highlight how change the estimates in this specific situation and how it can be improved increasing the number of cameras.

B. *Setup RW Model*

After the implementation of the virtual arena as reported above, we take into account the configuration of a realistic model, which would describe a real path of some person that walk in the created arena. In order to represent N_{target} targets which walk in the room, clearly without entering the area occupied by the obstacle, we adopted the RW model.

We consider the hidden state $\mathbf{x}_k \in \mathbb{R}^4$, which is composed of the 2D-position and the velocities of the target in both directions. Then, we have implemented the RW considering a linear model, with matrices:

$$\mathbf{F} = \begin{bmatrix} \mathbf{I}_2 & 0.5 \cdot T_S \cdot \mathbf{I}_2 \\ \mathbf{0}_2 & \mathbf{I}_2 \end{bmatrix} \qquad \mathbf{G} = \begin{bmatrix} 0.3 \cdot \mathbf{I}_2 \\ 0.1 \cdot \mathbf{I}_2 \end{bmatrix} \tag{19}$$

where $T_S = 1$ s. The dynamics of the target are described by the following equation:

$$\mathbf{x}(k+1) = \mathbf{F}\mathbf{x}(k) + \mathbf{G}\mathbf{w}(k) \tag{20}$$

where we suppose $\mathbf{w}(k)$ to be a white random noise with zero mean and variance $\mathbf{Q} = 0.5 \cdot \mathbf{I}_2$.

In the simulation we perform $N = 200$ steps all inside the arena, if the target touches the limits, he goes inside in order to obtain a full tracking and analyze the behaviour of the cameras.

A possible path of N steps of $N_{target} = 3$ that we have simulated can be seen in the following picture:

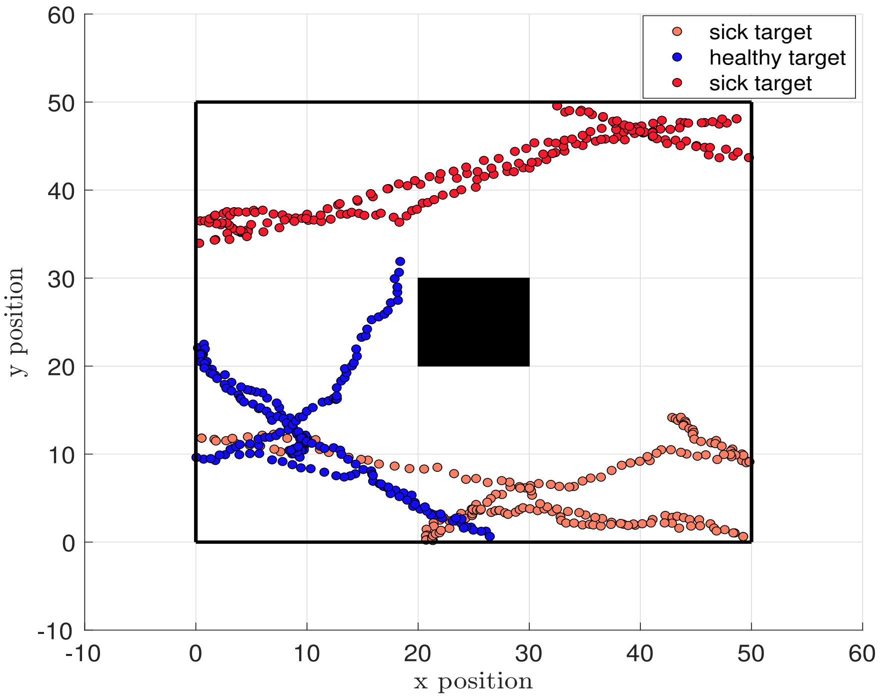

Figure 6. Example of random walks with 2 sick targets and one healthy

It is easy to note that in the Figure 6 are represented the real paths of each target. Each person is denoted by a different color, but more precisely the target represented by warm colors are sick, instead the others are healthy. In this particular case there are two sick targets and one healthy as described by the legend. The temperature of each target is given by the thermal cameras which we assumed that work perfectly. Thus this property of each target is given as a prior information, then the CEKF is applied only with the sick persons. For this reason we make the algorithm with lower computational burden.

C. Setup Observation Model

In this project, as it is explained in the introduction I, it is assumed to use depth cameras to measure the distance between the target and each camera. Clearly the hope is that this quantity is as accurate as possible, certainly it is affected by external and internal noises. Thus the observation model for our problem is designed as in the equation (6). The measure $h_k(x_k)$ denotes the Euclidean distance between a camera positioned in $(x_{cam}, y_{cam}) \in \mathbb{R} \times \mathbb{R}$ and a target placed in $(x_{tar}, y_{tar}) \in \mathbb{R} \times \mathbb{R}$. Clearly this quantity is not linear and it is described by the following equation:

$$h_k(x_k) = \sqrt{(x_{cam} - x_{tar})^2 + (y_{cam} - y_{tar})^2} \tag{21}$$

Furthermore, in the observation model described in (6) we introduce v_k, which is a white random noise with zero mean and variance $\mathbf{R} \in \mathbb{R}^2$ and it is a $J \times J$ matrix, where J is the number of cameras chosen for the network.

D. CEKF implementation

After the implementation of the environment, and the generation of the global model, we compute the CEKF as explained in the section IV. Hence, to obtain the estimation of the target position, it is adopted the prediction and filtering steps of the CEKF described by the equations (10)-(13). In both the two steps, we use the matrices $\mathbf{F}$ and $\mathbf{G}$ described in the equation (19). In particular we compute the matrix $\mathbf{H}_k$ performing the linearization introduced in the equation (7). Moreover we define the initial conditions for the CEKF as follow:

$$\mathbf{P}_{0|-1} = 100 \cdot \mathbf{I}_4 \qquad \mathbf{x}_{0|0} = [0, 0, 0, 0]^T \tag{22}$$

Hence, adopting this setup and the matrices just described, we use the information given by the thermal cameras to identify the sick target in order to estimate with CEKF the state x_k only of the sick persons. Thus we carry out different kinds of simulations and tests, to find the best solution and to analyze how the estimates can change adopting different arrangement of cameras network. In the next subsection it is explained more in detail the results obtained with the following simulations:

- CEKF in an ideal situation, and how the estimates change, using different quantity of cameras;
- CEKF in a real scenario, considering only the cameras which have an higher probability of detection;
- CEKF in a real scenario, weighting the measures of the cameras more reliable and removing the others;

1) Simulation of CEKF in an ideal situation: Adopting the setup and the matrices described above, the first simulation that it is computed, consists in considering all cameras of the network in an ideal situation. Therefore, it is assumed that all cameras have sure probability to detect a person, if the sick target walks in their FOV. Then any kind of detection error by the cameras is not considered and moreover it is adopted a matrix $\mathbf{R}$ with small values to represent the variance of the measure noise given by the equation (6). This choice is taken to make the situation ideal. Hence it is computed the estimate of the position of one sick target, adopting different quantity of cameras for the network as follow.

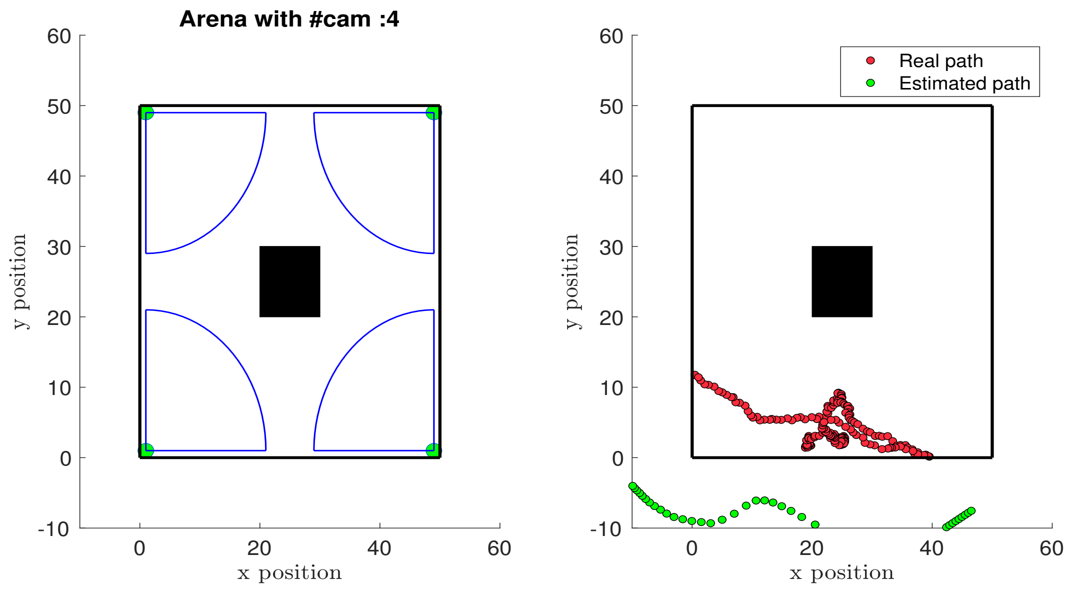

Figure 7. Ideal situation using 4 cameras

In the Figure 7, at the left it is shown the arrangement of the four cameras and at the right the estimate of the position of the sick target denoted by the green markers. It is easy to note that with four cameras, even if we adopt a covariance matrix small, the CEKF is not able to estimate accurately the position of the target, whose reason is explained more precisely in the section V related to the multi-tracking. Indeed with this camera network version, there are too arena's zones uncovered by the camera's FOV, in this way the sick person is positioned often in an unsupervised zone; moreover the absence of overlapping between camera's FOV causes further a bad estimate of the target position. Hence the algorithm is tested by increasing the number of cameras and by improving the arrangement of the network. Increasing the quantity of sensors we can obtain better results, as we can see in the following picture:

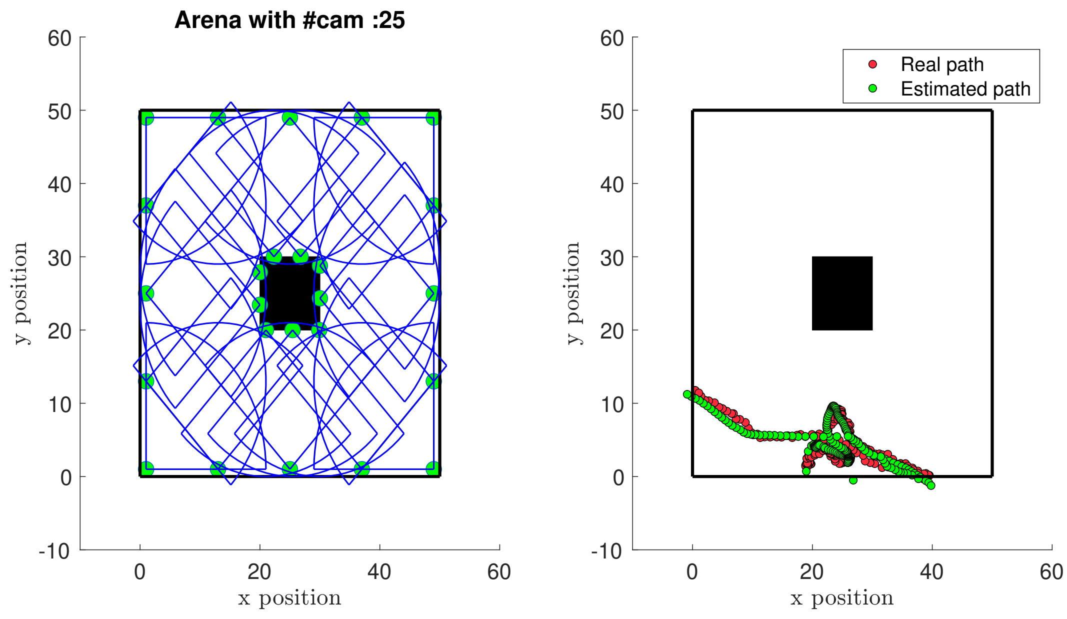

Figure 8. Ideal situation using 25 cameras

In Figure 8, clearly we get a better solution, indeed thanks to the overlapping of camera's FOV, the position estimate become more accurately, as reported in the section V. Only in the first steps the CEKF is not able to predict the correct position because it starts from the initial position described in the equation (22). Furthermore, if we try to increase even more the quantity of cameras we do not denote a performance improvement, this can be explained by the fact that having several (more than necessary) overlapping FOVs, the network does not add any useful information. Therefore, with this arena and by using more cameras the performance does not improve so much, instead the complexity and the bandwidth consumption increase significantly.

2) CEKF with a Bernoulli vector: In this section it is described the simulation and results obtained adopting the CEKF in a more realistic scenario. In order to do this, we implement a Bernoulli vector B during a training step, as reported in the section VI-A. It allows to identify the reliability of each camera and its probability of detection. After the training step we enhance the bad behaviour of each unreliable camera. In the simulation we increase the measurement noise, given by the covariance matrix

R. In particular if the i^{th} camera is defined unreliable by the vector B, we increase the correspondent component $r_i = [tr(\mathbf{R})]_i$.
In the following picture is reported the CEKF estimate considering all 36 cameras, including the unreliable sensors.

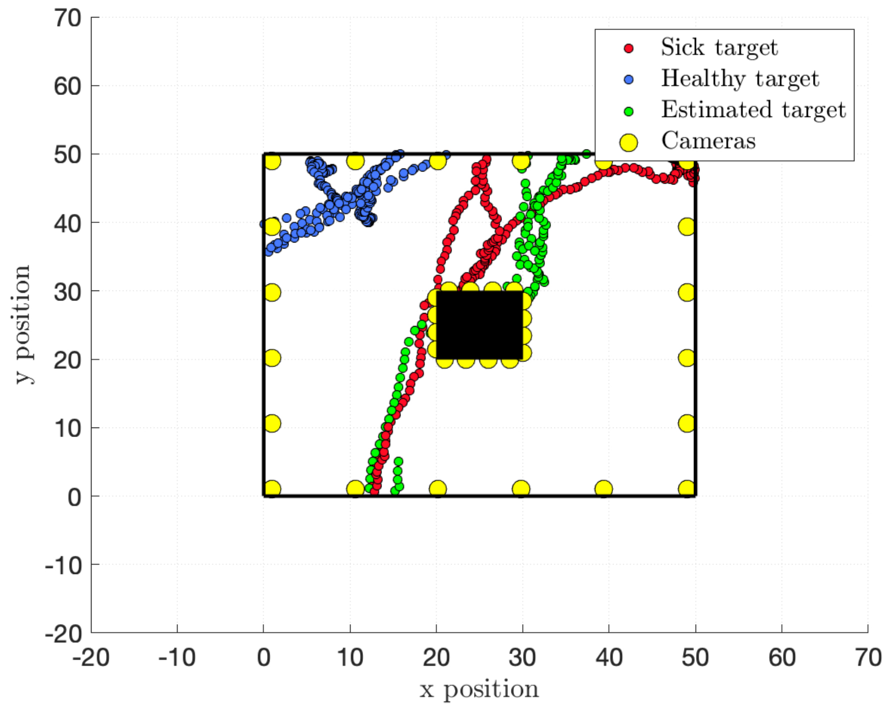

Figure 9. Estimate considering all cameras

In this specific example, there are 34 cameras that work well, and 2 that work not so good. Hence, as we expected, the estimated path of the sick target, denoted by the green markers, is wrong because the distances given by the two bad sensors are completely affected by noise. So when the target is positioned in the bad camera's FOV, the CEKF is not able to estimate correctly person's position. In order to solve this problem, it is tried to use the Bernoulli vector as a prior information, in order to completely remove the cameras that do not work. In this way, the only information that are used for the estimate by the CEKF are the distances measured by only the reliable cameras, in such a way to make the estimate as less affected as possible by error.

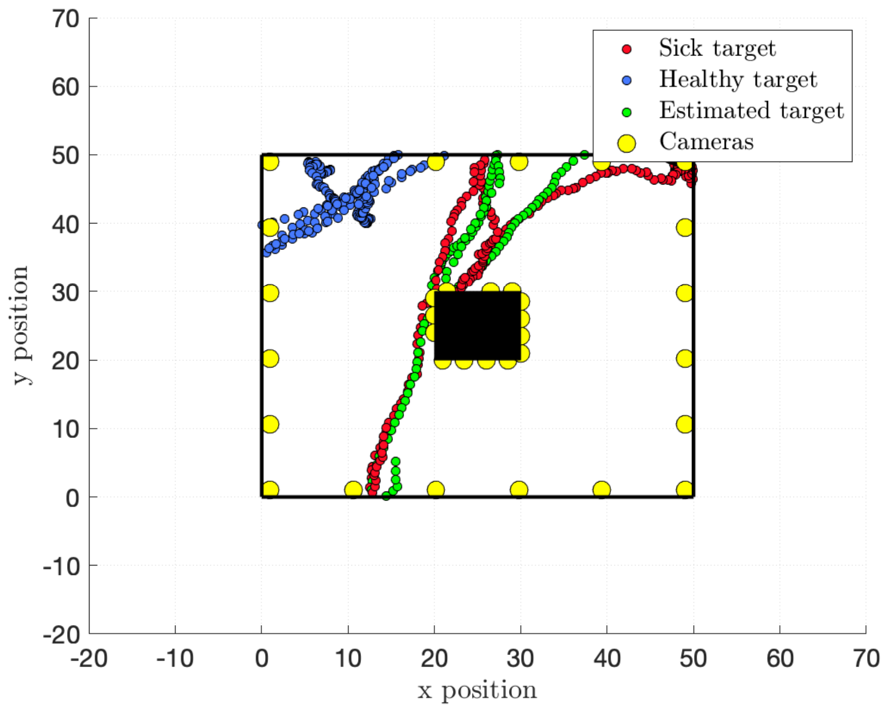

Figure 10. Estimate considering only reliable cameras

Clearly, as it can be seen in Figure 10, only the 34 reliable cameras are considered. Moreover, it is easy note that the estimated path is improved, because the information affected by errors are not evaluated. In this way, the matrix **R** that describes the error of each measure has reduced sizes, because the i_{th} components correspondent to i_{th} camera which does not work, is removed. It is also important to remark that the estimate could not be perfect, because by removing some sensors there is a risk to cover less area and to fall in the case in which any camera can see the target. This problem is reported more in detail in the section IX, related to possible improvements.

3) Weighted CEKF: Later, it is tried to improve further the estimate of target's position, then it is implemented a weighted version of the CEKF, as described in section VI-B. In the following Figure is reported the estimate obtained adopting the WCEKF.

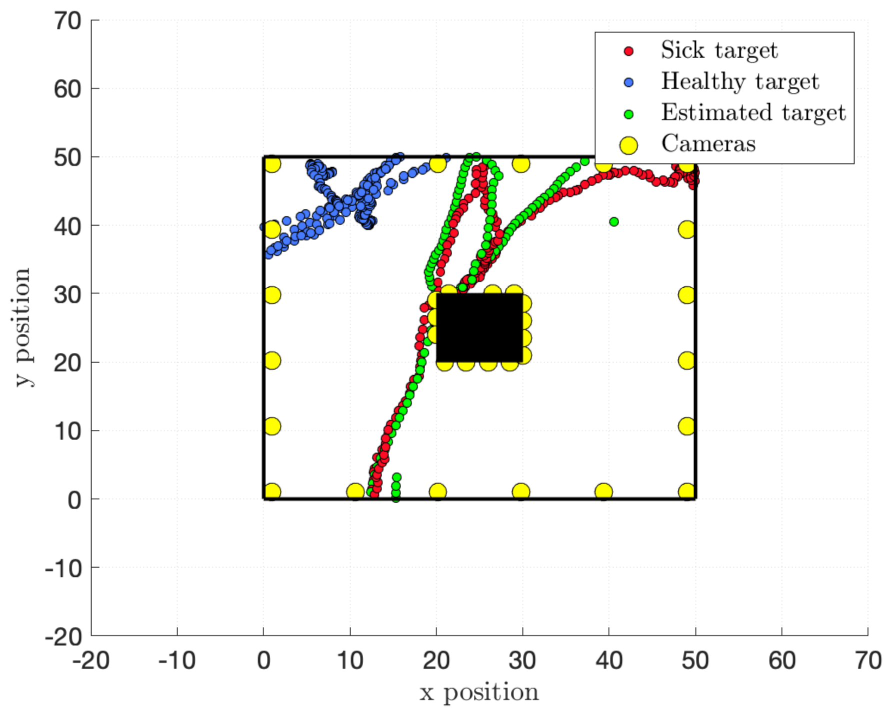

Figure 11. Estimate considering only weighted reliable cameras

Even in this version, the unreliable cameras are removed. Furthermore the information given by the reliable sensors are weighted, according to their detection's probability obtained in the training step. Clearly, it can be noticed that even in this case the estimate is better than the ones obtained considering all cameras, Figure 9, because it is considered only the reliable cameras. Moreover, it can be remarked that adopting this method it is obtained the best results, because we minimize the influence on the estimate of less reliable cameras.

VIII. Conclusion

In the design phase of this project we analyzed different approaches in order to deal with possible issue of camera networks. Moreover, we wanted to insert a realistic implementation of the proposed architecture because our aim was not only to find a theoretical solution but also to propose a new solution for the containment of the pandemic with heuristic solutions.

Implementing the CEKF algorithm and observing all the results obtained in our simulation, it is reasonable to assume that this approach can represent a suitable good starting point. Considering that our sensors measure the distances, it allows us to avoid a calibration step in the configuration of the networks. Hence, this approach can be implemented quickly in any environment. However, our system is not so efficient in terms of computational complexity and stability, indeed if the central processing unit breaks, the entire network system stopping to work. Possible improvements are presented in the next section IX.

At each time instant we acquire data only from reliable cameras that can see the target, implementing the Bernoulli estimation we can obtain better results than taking into account all cameras. Another enhancement is the WCEKF, it allows us to reach better results when we are considering a powerful network cameras with many sensors. As matter of fact, reducing the number of cameras the information available is poor. Then, weighting the measurement of each camera we obtain a worse solution.
Moreover, we think that this could more useful when the weights are estimated in real time or with a calibration procedure. Ideally, weighting the sensor measurements this improvement leads us to have a more robust estimator.

IX. Possible improvements

Studying and discussing the camera network problem we realize how much this topic is subjected to continuous improvements, and how many under problems it can hide.

A. DEKF and Particle Filters

At the beginning we wanted to implement the DEKF, we studied different solutions but we was not able to develop a suitable simulation using these approaches because they are based on methods that we did not study. If we had more time, we would have been willing to study this new algorithms, like the ADMM, then deliver a more research project. Therefore, we simplify the problem considering a CEKF even because it is not only a theoretically solution but it is really implemented in the real world.
Another solution could be the use of particle filters, in this way we could not consider the gaussianity for the state and the output noise. As reported in [10], particle filters is more potential useful in applications with non-linear model or non-Gaussian noise.

B. Acquire more data from cameras - PTZ camera

In this report we assume to obtain from a single camera only the distance, but most of the camera tracking methods, implemented in the real world, are based on powerful computer vision algorithms. Moreover, the performance of dynamic scene algorithms often suffers because of the inability to effectively acquire features on the targets, particularly when they are distributed over a wide FOV. A suitable solution is to rotate the single camera in order to cover as much area as possible. This can be achieved by dynamically controlling the parameters of a Pan Tilt Zoom (PTZ) camera network. In this way we can maximize the network resources reducing costs and hardware requirements. As described by [11], in order to track one person obtaining a high-resolution image of him, it is necessary to change dynamically the parameters of a camera. Therefore, we need to use a cooperative controller because each camera's parameter setting entails certain constraints on other cameras. For example, if a camera zooms in to the face of one particular person, thus narrowing its FOV, it risks losing much of that person and the surroundings, therefore being less useful for target tracking. Another camera can compensate for this by adjusting its parameters. This requires analysis of the video data in a collaborative network-centric manner, leading to a cost-effective method to obtain high-resolution images for features at dynamically changing locations.

References

[1] J. Rangel, S. Soldan, and A. Kroll, “3d thermal imaging: Fusion of thermography and depth cameras.” [Online]. Available: https://www.ndt.net/article/qirt2014/papers/QIRT-2014-035.pdf

[2] R. E. Kalman, “A new approach to linear filtering and prediction problems,” *Journal of Basic Engineering*, vol. 82, p. 35–45, 1960.

[3] B. D. O. Anderson and J. B. Moore, “Optimal Filtering,” *EnglewoodCliffs NJ: Prentice-Hall*, 1979. [Online]. Available: http://users.cecs.anu.edu.au/~john/papers/BOOK/B02.pdf

[4] S. Wang and A. Dekorsky, “Distributed Consensus-Based Extended Kalman Filtering: A Bayesian Perspective,” 2019 27th European Signal Processing Conference (EUSIPCO), A Coruna, Spain, 2019, pp. 1-5.

[5] T. D. Räty, “Survey on contemporary remote surveillance systems for public safety,” *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, vol. 40, no. 5, pp. 493–515, 2010.

[6] R. Arroyo, J. J. Yebes, L. M. Bergasa, I. G. Daza, and J. Almazán, “Expert video-surveillance system for real-time detection of suspicious behaviors in shopping malls,” *Expert Systems with Applications*, vol. 42, no. 21, pp. 7991–8005, 2015.

[7] Z. Chen, “Bayesian filtering: From kalman filters to particle filters, and beyond,” *Statistics*, vol. 182, 2003.

[8] S. Särkkä, “Bayesian filtering and smoothing,” *Institute of Mathematical Statistics Textbooks. Cambridge: Cambridge University Press*, 2013.

[9] A. V. Kurilkin and S. V. Ivanov, “A comparison of methods to detect people flow using video processing,” *Procedia Computer Science*, vol. 101, pp. 125–134, 2016, 5th International Young Scientist Conference on Computational Science, YSC 2016, 26-28 October 2016, Krakow, Poland.

[10] Y. Xuebing, P. Jingui, and Y. Xuebing, “Particle filters for tracking target with a camera,” in *2007 IEEE International Conference on Control and Automation*, 2007, pp. 2248–2251.

[11] C. Ding, B. Song, A. Morye, J. A. Farrell, and A. K. Roy-Chowdhury, “Collaborative sensing in a distributed ptz camera network,” *IEEE Transactions on Image Processing*, vol. 21, no. 7, pp. 3282–3295, 2012.