

Article

Optimization of Camera and Radar Placement for Sensor Fusion and Ball Tracking in Sports

Dylan Kamstra * and Johan Pieter de Villiers 

Department of Electrical, Electronic and Computer Engineering, University of Pretoria,
Pretoria 0002, South Africa; pieter.devilliers@up.ac.za

* Correspondence: u19090634@tuks.co.za

Abstract

The placement of sensors in an environment can significantly impact the sensing performance of a sensor fusion system. In this paper, the placement of cameras and radars is optimized based on the log determinant of the fused measurement noise of the sensor measurements. This is achieved by mapping the measurements into 3D Cartesian space and applying covariance intersection to obtain a final measurement distribution, which is taken as the measurement noise. The method was tested against random initial placements and optimization runs of sensors for a system that is intended for ball tracking in sports. The particular use case involves the tracking of a cricket ball for the purpose of match evaluation and assisted umpiring. However, in principle, the method is applicable to any sensor placement problem in which the objective is localization and tracking. The results indicate an improved root mean squared error for the optimized sensor placements, which in turn implies a reduction in the measurement noise covariance.

Keywords: covariance intersection; triangulation; sensor placement; optimization

1. Introduction

Optimal sensor placement is a vital aspect of system design in any system where sensor measurements are used for any kind of inference, such as state estimation or classification. This is especially true in sensor fusion systems such as cameras, radars, and lidars, as the placement of these types of sensors affects the characteristics of the measurements, in the form of the sensors' resolution, potential occlusions, coverage of the desired area, and redundancy. In addition, some sensor types may complement the weaknesses of other sensor types. In many circumstances, the placement of sensors has been left to expert human judgment, yet this is subject to the expert's experience, which can be misguided.

Common applications of optimal sensor placement are attempts to increase the field of view (FOV) of a given area constrained by the number and type of cameras [1–3]. This type of application has a well-defined objective; however, the problem is NP-hard as there are many placements of the cameras that can yield the same result. An analogous problem to maximizing the FOV is tracking indoor movement [4]. These applications look at a simplified map of the region of interest (ROI), where while the sensors are modelled in 3D, the optimization is done in a 2D bird's-eye view (BEV) of the ROI, which ties into their objective function.

In many applications, objects of interest are triangulated from their observations in images, most notably in tracking-based tasks, to obtain valuable 3D state information about the object. This can be used for reconstruction purposes, such as in motion capture [5].



Academic Editor: Ning Xiong

Received: 6 March 2026

Revised: 13 April 2026

Accepted: 15 April 2026

Published: 30 April 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

Here, cameras are allowed to vary within a constrained range, limited by the nature of the problem. This can be extended to the optimal view of human pose [6]. Furthermore, the problem of triangulation involves the problem of overlapping coverage of camera sensors [7], which is important to consider in a static environment and influences the number of cameras used.

A different approach to sensor placement is to work with the covariance of the sensor measurements. The aim is to decrease the volume of the ellipsoid of the measurement covariance matrix. In [8], the problem is reformulated as a sensor selection problem, where the measurement function is linear in the state space. The log determinant of the covariance matrix is used as the minimization criteria to select the correct sensors. Alternatively, the trace of the steady-state error covariance of a Kalman filter (KF) can be used to place the sensors in nodes in a graph [9,10].

To optimize camera placement, the most commonly used algorithms do not have access to the closed-form expression of the objective function derivative with respect to sensor pose. As such, algorithms such as Simulated Annealing (SA) [5,11], Particle Swarm Optimization (PSO) [12], and genetic algorithms (GA) [7] are used, which explore the surface of the loss functions. Greedy-based algorithms are another approach [13], where sensors are placed based on a voting scheme. Although these algorithms produce good results, they suffer from large computational loads and long execution times. An alternative approach is to use Gaussian processes (GP) [14] to find sensor configurations that maximize the mutual information between the chosen sensor locations and the locations not chosen. Not all the sensors need to be chosen at the start, as in [15], where sensors are added one by one to ensure that the minimum number of sensors is used.

In [16], error propagation is presented for computer vision problems, including curve fitting, local feature extraction, and exterior and relative orientation. However, it does not address the problem of triangulation. In [17], an algorithm is given to find pairs of points to obtain an initial triangulation, which can then be used to improve triangulations from multiple sensors. Many attempts have been made to find a fast, robust three-view camera triangulation algorithm [18–20], yet these are limited in the number of cameras that they can use, a number which is baked into the problem to be solved.

In this paper, a Gradient Descent (GD) algorithm is presented for minimizing a matrix norm of the measurement covariance (obtained by covariance intersection) for a system of cameras and radars by varying the position and orientation of the sensors. The measurement noise dominates the accuracy of tracking filters; minimizing the measurement noise will increase the performance of a filter that uses the sensors. This is achieved by fusing the measurements of the sensors in Cartesian space based on their inherent noise parameters. This is done through the use of covariance intersection (CI) [21]. CI has been used to fuse cameras and radars/lidars in simultaneous localization and mapping (SLAM) [22–24], as the cross-correlation between the sensor types is unknown.

The remainder of this paper is structured as follows. Section 2 provides a mathematical objective to the problem and presents the assumptions made. In Section 3, the sensor models and the fusion of their measurements are presented, while in Section 4 the basic GD algorithm is formulated and the constraints of the system are presented. In Section 5, the system is tested for different initial placements of the sensors, and the results are discussed. Lastly, Section 7 presents the conclusions and potential future work.

2. Problem Formulation and Assumptions

Given a space $\mathbb{R}^3$ or set of coordinates in the space that must be observed by at least two sensors, either a camera and/or a radar, the problem can be formulated as finding the optimal sensor centre in the world coordinates c and orientation transformation matrix

$\mathbf{R}$, which transforms a state-space vector from world coordinates to sensor coordinates, where $\mathbf{c}$ is a point-like 3×1 vector representation of the sensor's position (the focal point of the camera or the antenna location of the radar), while $\mathbf{R}$ is a 3×3 rotation matrix. Each sensor's $\mathbf{c}$ and $\mathbf{R}$ are initialized from an initial set of offsets and orientations, $\bar{\mathbf{c}}$ and $\bar{\mathbf{R}}$, such that a matrix norm (the determinant, trace, or root mean square error (RMSE)) of the fused measurement covariance is minimized. The problem can be formulated as

$$\mathcal{C}_{opt}, \mathcal{R}_{opt} = \operatorname{argmin} \frac{1}{N} \sum_{\mathbf{x}_i \in \mathcal{X}} \operatorname{norm}[A(\mathcal{Y}(\mathcal{C}, \mathcal{R}, \mathbf{x}_i), \mathcal{C}, \mathcal{R})] \quad (1)$$

where $\mathcal{X} = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N\}$, $\mathbf{x}_n$ is a 3×1 position vector of a target/object, or a point of interest (POI), that is observed, for example, at sample time n . The measurement of $\mathbf{x}_n$ is given by $\mathcal{Y}$, which produces a set of measurement vectors $\{\mathbf{y}_n^{(1)}, \mathbf{y}_n^{(2)}, \dots, \mathbf{y}_n^{(I)}\}$, where I is the number of sensors, and $\mathbf{y}_n^{(i)}$ is a $M \times 1$ vector of measurements of $\mathbf{x}_n$ made by the i^{th} sensors. The set of offset vectors for the sensors is given by $\mathcal{C} = \{\mathbf{c}^{(1)}, \mathbf{c}^{(2)}, \dots, \mathbf{c}^{(I)}\}$, $\mathcal{R} = \{\mathbf{R}^{(1)}, \mathbf{R}^{(2)}, \dots, \mathbf{R}^{(I)}\}$ is the set of sensor rotation matrices, and A is the measurement fusion function that produces a 3×3 covariance matrix.

The algorithm for the measurement covariance intersection is discussed in Section 3.5, which shows how it is related to the rotation and position of each of the sensors; however, it is not difficult to see that the position and orientation will have an effect on the outcome of the fusion. It is important to note that $\mathbf{c}_0$ and $\mathbf{R}_0$ has the potential to play a vital role in the final values of $\mathcal{C}$ and $\mathcal{R}$, depending on the number of sensors used, as this is an NP problem.

This formulation allows for an arbitrary number of sensors greater than two to be used, as a minimum of two sensors is required for the intersection of the measurements' covariances to have a well-defined covariance matrix. However, this approach makes a basic assumption: that all the projections of the POIs in $\mathbf{X}$ onto the sensors will result in a successful fusion. This assumption is intuitive if the environment in which the fusion will occur is in an ROI, i.e., it is known that all the fusion will occur within a bounded area. Since two sensors cannot be in the same place, there will always be an information exchange between the sensors.

3. Covariance Fusion Theory

3.1. Camera Mapping

A camera can be modelled as having both intrinsic and extrinsic parameters. The extrinsic parameters are its position in the world coordinates $\mathbf{c}$ and rotation $\mathbf{R}$. The intrinsic parameters are given by

$$\mathbf{K} = \begin{bmatrix} f_x & s & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (2)$$

where f_x and f_y are the camera focal points for the x -axis and y -axis, s is the skew parameter, and u_0 and v_0 are pixel offsets so that the origin of the image is at the top-left corner. For the remainder of the paper, it is assumed that $\mathbf{K}$ is kept constant. This is set as the focal length and the translation of the camera are intertwined, i.e., moving the camera backwards while increasing the focal length will negate each other; thus, keeping it constant removes it from the dynamic variables. This fact will come into use when constraining the offset of the cameras.

The complete mapping of a POI $\mathbf{x} = \begin{bmatrix} x & y & z \end{bmatrix}^T$ onto the image is given by

$$\lambda \begin{bmatrix} \mathbf{u} \\ 1 \end{bmatrix} = \mathbf{K}[\mathbf{R} | -\mathbf{R}\mathbf{c}] \begin{bmatrix} \mathbf{x} \\ 1 \end{bmatrix} = \mathbf{K}(\mathbf{R}(\mathbf{x} - \mathbf{c})), \quad (3)$$

where the pixels are defined up to a scale factor λ , $\mathbf{u}^T = \begin{bmatrix} u & v \end{bmatrix}^T$ are the position of the pixel, and homogeneous coordinates are used for the 3D POI and pixels. It is the division of λ in (3) that moves the POI $\mathbf{u}$ from the 3D domain to the 2D-pixel domain. The use of homogeneous coordinates in $\mathbf{x}$ is such that the mapping can be combined into a matrix $\mathbf{P}$. Expanding (3) out and incorporating the division by λ , one obtains

$$\mathbf{u} = \begin{bmatrix} \frac{\mathbf{P}_1[\mathbf{x}^T \ 1]^T}{\mathbf{P}_3[\mathbf{x}^T \ 1]^T} & \frac{\mathbf{P}_2[\mathbf{x}^T \ 1]^T}{\mathbf{P}_3[\mathbf{x}^T \ 1]^T} \end{bmatrix}^T, \quad (4)$$

where $\mathbf{P}_i$ is the i th row of $\mathbf{P}$, given by

$$\mathbf{P} = \mathbf{K}[\mathbf{R} | -\mathbf{R}\mathbf{c}]. \quad (5)$$

Equation (4) shows how important $\mathbf{R}$ and $\mathbf{c}$ are in the mapping process.

3.2. Radar Mapping

Similar to the camera, the radar is defined in space by a rotation and a translation. The mapping of a POI $\mathbf{x}$ from world coordinates to radar coordinates $\mathbf{r} = (r, \theta)$, where r is the range and θ is the azimuth, is a non-linear mapping of

$$r = \|\mathbf{R}(\mathbf{x} - \mathbf{c})\|, \quad (6)$$

$$\theta = \arctan \frac{\mathbf{R}_2(\mathbf{x} - \mathbf{c})}{\mathbf{R}_1(\mathbf{x} - \mathbf{c})}, \quad (7)$$

where $\mathbf{R}_1$ and $\mathbf{R}_2$ are the first and second rows of the rotation matrix $\mathbf{R}$ respectively. This mapping goes from 3D Cartesian coordinates to 2D polar coordinates, where the elevation of the POI is lost.

For completeness, the elevation of a POI is given by

$$\rho = \arccos \frac{\mathbf{R}_3(\mathbf{x} - \mathbf{c})}{r}. \quad (8)$$

3.3. Camera Covariance

The mechanism by which noise enters the camera measurement process in (3) is the result of several influences. Even if the calibration of the camera is performed perfectly, the quantization of the light intensities owing to pixelization and noise in the electronics of the camera at the moment an image is taken will perturb the pixel values. We defer to the central limit theorem, and make a simplifying assumption where the addition of several measurement imperfections results in an additive Gaussian noise model; therefore, the perturbed pixel value $\tilde{\mathbf{u}}$ is given by

$$\tilde{\mathbf{u}} = \mathbf{u} + \mathbf{w}, \quad (9)$$

where $\mathbf{w}$ is zero-mean independent Gaussian noise with covariance matrix Σ_w . Since cameras lose depth information, when projecting the pixel point back into 3D space, the coordinate $\mathbf{x}$ exists along the ray

$$\mathbf{x} = \lambda \mathbf{R}^T \mathbf{K}^{-1} \begin{bmatrix} \tilde{\mathbf{u}} \\ 1 \end{bmatrix} + \mathbf{c}. \quad (10)$$

If the value of λ is known, then the 3D POI can be estimated. Equation (10) represents an affine transformation of the pixel point, which can be applied to a Gaussian distribution; however, care must be taken in defining the values in Σ .

Consider the Gaussian distribution

$$\mathbf{g} = \mathcal{N}\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix}_g, \mathbf{\Omega}_g^{-1}\right),$$

where $\mathbf{\Omega}_g$ is a 3×3 precision matrix given by

$$\mathbf{\Omega}_g = \begin{bmatrix} \Sigma_g^{-1} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix},$$

and Σ_g is a 2×2 covariance matrix, and $\mathbf{0}$ is a 2×1 zero vector. The Gaussian distribution described by $\mathbf{g}$ can be seen as a 2D Gaussian distribution projected into 3D space, where the variance along the z -axis approaches infinity. This is why it is easier to work with the precision matrix, as the covariance matrix is undefined—this can be seen by noting that the determinant of the precision matrix is zero, and so its inverse matrix does not exist.

Now consider a similarly defined Gaussian distribution

$$\mathbf{h} = \mathcal{N}\left(\begin{bmatrix} x \\ y \\ z \end{bmatrix}_h, \mathbf{\Omega}_h^{-1}\right),$$

where $\mathbf{\Omega}_h$ is defined as

$$\mathbf{\Omega}_h = \begin{bmatrix} 0 & \mathbf{0}^T \\ \mathbf{0} & \Sigma_h^{-1} \end{bmatrix}.$$

This Gaussian distribution similarly has zero precision along the x -axis. Looking at the two Gaussian distributions $\mathbf{g}$ and $\mathbf{h}$, it is clear that together they constrain the covariance matrix along all axes of 3D space, $\mathbf{g}$ constrains the variance in the xy plane and $\mathbf{h}$ constrains the variance in the yz plane. Taking the product of these two functions yields a Gaussian distribution of

$$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma}'),$$

where $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are given by

$$\begin{aligned} \boldsymbol{\Sigma}' &= (\mathbf{\Omega}_g + \mathbf{\Omega}_h)^{-1} \\ &= \begin{bmatrix} \Omega_g^{[11]} & \Omega_g^{[12]} & 0 \\ \Omega_g^{[21]} & \Omega_g^{[22]} + \Omega_h^{[11]} & \Omega_h^{[12]} \\ 0 & \Omega_h^{[21]} & \Omega_h^{[22]} \end{bmatrix}^{-1}, \\ \boldsymbol{\mu} &= \boldsymbol{\Sigma}' \left(\mathbf{\Omega}_g \begin{bmatrix} x \\ y \\ - \end{bmatrix}_g + \mathbf{\Omega}_h \begin{bmatrix} - \\ y \\ z \end{bmatrix}_h \right), \end{aligned}$$

where $\Omega^{[ij]}$ represents the element in the i th row and j th column of $\mathbf{\Omega}$, and $(-)$ is used to show that the value of this variable can be set arbitrarily as the corresponding values in the precision

matrix are zero. Looking at Σ' , it is clear that this matrix is well-defined, and that μ contains no redundant information. This can be scaled to an arbitrary number of low-dimensional Gaussian distributions projected up into the same higher-dimensional space.

Within the context of camera sensors, the ray coming from the camera center through the image point (given by applying the inverse intrinsic matrix to the pixel point) can be viewed in a similar light as projecting a 2D Gaussian distribution in the camera pixel space into the 3D world coordinates space, where the depth axis has zero precision.

In camera coordinates, the depth axis is given by the z -axis. The 2D Gaussian distribution is

$$\mathcal{N}\left(\begin{bmatrix} u \\ v \end{bmatrix}, \Omega_u^{-1}\right), \quad (11)$$

where Ω_u is the pixel precision matrix, the image coordinates in 3D are given by

$$\mathcal{N}\left(\mathbf{K}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}, \Omega_x^{-1}\right), \quad (12)$$

and

$$\Omega_x = \mathbf{K}_2^T \begin{bmatrix} \Omega_u & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} \mathbf{K}_2, \quad (13)$$

where the intrinsic parameter matrix results in addition owing to u_0 and v_0 in (2). This results in a translated covariance matrix in (13) which violates the fact that covariance matrices are only scaled and rotated. To resolve this, the upper left 2×2 matrix in $\mathbf{K}$ is used and the remaining elements are set to zero, denoted $\mathbf{K}_2$.

Since the Gaussian is formulated in terms of precision, the matrices in (13) are multiplied using the identity

$$(\mathbf{A}_1 \mathbf{A}_2 \cdots \mathbf{A}_{k-1} \mathbf{A}_k)^{-1} = \mathbf{A}_k^{-1} \mathbf{A}_{k-1}^{-1} \cdots \mathbf{A}_2^{-1} \mathbf{A}_1^{-1},$$

where $\mathbf{K}^{-1}$ is brought into the matrix inversion in (13).

The formulation in (12) is, however, incomplete. In (12), the line that has an undetermined variance is the camera's z -axis, not the ray defined in (10). To correctly align the Gaussian distribution, an additional rotation is applied to the precision matrix. Let the projected coordinate on the image plane in 3D space be defined as

$$\hat{\mathbf{x}} = \frac{\mathbf{K}^{-1} \begin{bmatrix} u & v & 1 \end{bmatrix}^T}{\|\mathbf{K}^{-1} \begin{bmatrix} u & v & 1 \end{bmatrix}^T\|}. \quad (14)$$

The rotation from $\mathbf{x}^\circ$ to the z -axis basis function $\mathbf{e}_z = \begin{bmatrix} 0 & 0 & 1 \end{bmatrix}^T$, i.e., $\mathbf{R}_{\hat{\mathbf{x}}} \hat{\mathbf{x}} = \mathbf{e}_z$, can be constructed by calculating the correct basis vectors. The new z -axis is given by

$$\mathbf{e}'_z = \frac{\hat{\mathbf{x}}}{\|\hat{\mathbf{x}}\|}.$$

The new basis vector for the x -axis is constructed by making $\mathbf{e}_x$ perpendicular to the new z -axis basis function by

$$\mathbf{e}'_x = \frac{\mathbf{e}_x - (\mathbf{e}_x \cdot \mathbf{e}'_z) \mathbf{e}'_z}{\|\mathbf{e}_x - (\mathbf{e}_x \cdot \mathbf{e}'_z) \mathbf{e}'_z\|}.$$

Lastly, the new y -axis basis is found by taking the cross-product of the two new basis functions

$$\mathbf{e}'_y = \mathbf{e}'_z \times \mathbf{e}'_x.$$

The rotation matrix is then defined as

$$\mathbf{R}_{\hat{\mathbf{x}}} = \begin{bmatrix} (\mathbf{e}'_x)^T \\ (\mathbf{e}'_y)^T \\ (\mathbf{e}'_z)^T \end{bmatrix}, \quad (15)$$

This formulation only allows for the Gaussian to be pitched and yawed, which will preserve the orientation of the covariance ellipse of the pixel noise, where the covariance ellipse or ellipsoid is defined as the N sigma constant covariance contour.

Applying this rotation matrix to (12), we get

$$\mathcal{N}(\mathbf{K}^{-1} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}, (\mathbf{R}_{\hat{\mathbf{x}}}^T \mathbf{K}_2^T \begin{bmatrix} \Omega_u & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} \mathbf{K}_2 \mathbf{R}_{\hat{\mathbf{x}}})^{-1}), \quad (16)$$

which now has the major axis of the covariance ellipsoid aligned with the ray through the camera center and $\hat{\mathbf{x}}$. When the rotation matrices enter into the matrix inversion, the rotation matrix property of its inverse, being its transpose, is used. Lastly, the rotation of the camera and the depth scale λ are applied to obtain the final precision matrix of

$$\Omega_c = \frac{1}{\lambda^2} \mathbf{R}^T \mathbf{R}_{x_n}^T \mathbf{K}_2^T \begin{bmatrix} \Omega_u & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} \mathbf{K}_2 \mathbf{R}_{x_n} \mathbf{R}, \quad (17)$$

which yields a Gaussian distribution of

$$\mathcal{N}(\mathbf{c}, (\Omega_c)^{-1}), \quad (18)$$

where the mean of the Gaussian is replaced by the camera center. This is possible as the Gaussian distribution is now in world coordinates, and the mean can be any value along the ray in (10). Setting it to the camera center allows for the covariance matrix to describe any ray coming from the camera center if required.

It is possible to combine $\mathbf{R}$ and $\mathbf{R}_{\hat{\mathbf{x}}}$ into a single rotation by applying $\mathbf{R}$ in (14); however, care must be taken to ensure that the orientation of the camera axes will not change. This is only a concern if the pixel covariance is non-isotropic.

While the Gaussian in (18) will provide the correct covariance, λ is still not known. To find it, two of these covariance ellipsoids from different cameras can be intersected using an arbitrary λ , where the resulting Gaussian mean is a triangulated coordinate. From this coordinate, λ can be found by projecting the coordinate back onto each camera. The now correctly scaled covariance ellipsoids can then be intersected again to get a new mean and covariance. This can be performed iteratively to refine the results, similar to the iterative triangulation algorithms in [25].

While this process does provide a triangulated mean, the intersected covariance is an intersection of the individual camera covariance ellipsoids, which in turn approximate conic distributions. This can be seen in Figure 1. A particle approach is given in [26,27]; however, it does not yield a neat closed-form 3D probability distribution. The presented approach is able to easily scale to many cameras, as the process of finding the point of intersection of the different rays is handled through the use of the Gaussian distributions.

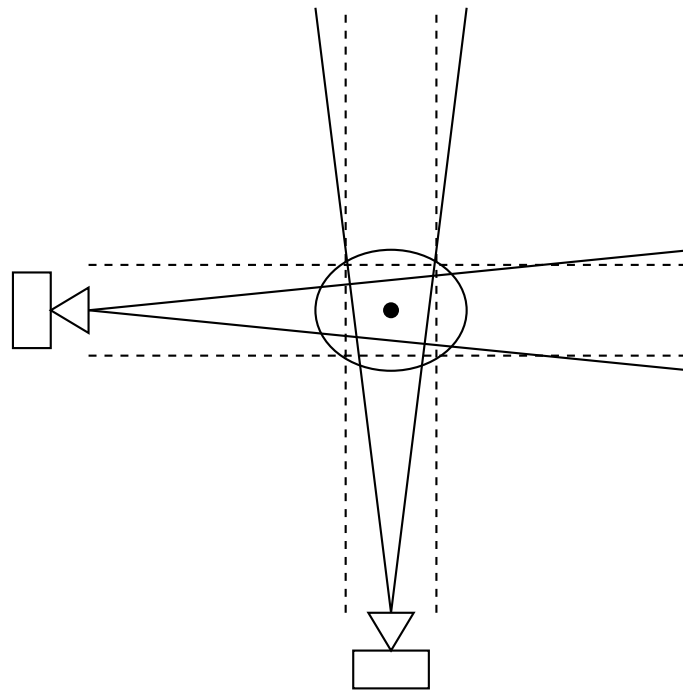


Figure 1. An exaggerated top-view of two cameras triangulating a POI, given by the circle. The solid lines show the one standard deviation ellipse, while the dashed lines are the approximation of the conic distributions. The resulting ellipse is shown.

3.4. Radar Covariance

The conversion of a radar measurement to Cartesian space poses a greater challenge. The type of automotive radar used in this study only obtains two of the three spherical coordinates, namely range and azimuth. Similar to the cameras λ , the elevation of a POI can be found iteratively once a mean coordinate has been found by multiple sensor covariance intersection, using an initial guess of the elevation, such as the mean elevation of the POIs.

Since the range and azimuth measurements contain some noise, the unbiased converted measurements (UCM) approach in [28] is used to convert the radar measurements to 2D Cartesian coordinates. The assumed elevation ρ is then used to define a radar POI coordinate in 3D space,

$$\dot{\mathbf{p}} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} e^{\sigma_\theta^2 r/2} \cos \theta \sin \rho \\ e^{\sigma_\theta^2 r/2} \sin \theta \sin \rho \\ r \cos \rho \end{bmatrix}, \quad (19)$$

where r and θ are the mean range and mean azimuth respectively, and σ_θ is the standard deviation of the azimuth. The inclusion of the scaling term is to remove the induced bias in the conversion between polar and Cartesian coordinates.

Similar to the camera's initial covariance, the radar's covariance has zero precision along the z -axis for an elevation of 0, that is, the covariance is constrained to the xy -plane. When the POI has a non-zero elevation, this plane needs to be tilted accordingly. This is accomplished by applying a rotation to the covariance matrix, which must ensure that the covariance matrix keeps the covariance between the x -axis and y -axis unchanged.

To achieve this, the axis is rotated to include the elevation, but also to set the azimuth of the measurement to 0. With an azimuth of 0, the x -axis must point towards the point $\dot{\mathbf{p}}$

$$\mathbf{e}'_x = \frac{\dot{\mathbf{p}}}{\|\dot{\mathbf{p}}\|}.$$

The y -axis is chosen to remain on the xy -plane and is made perpendicular to $\mathbf{e}'_x$.

$$\mathbf{e}'_y = \frac{\begin{bmatrix} -\sin \theta & \cos \theta & 0 \end{bmatrix}^T}{\left\| \begin{bmatrix} -\sin \theta & \cos \theta & 0 \end{bmatrix}^T \right\|}.$$

Lastly, the z-axis is found using the cross product.

$$\mathbf{e}'_z = \mathbf{e}'_x \times \mathbf{e}'_y,$$

and the full rotation matrix, which applies the elevation tilt, is given by

$$\mathbf{R}_{\hat{\mathbf{p}}} = \begin{bmatrix} (\mathbf{e}'_x)^T \\ (\mathbf{e}'_y)^T \\ (\mathbf{e}'_z)^T \end{bmatrix}. \quad (20)$$

In this new coordinate frame, the UCM variance for a Gaussian noise distribution is given by

$$\begin{aligned} R^{[11]} &= \frac{1}{2}(r^2 + \sigma_r^2)[1 + e^{-2\sigma_\theta^2}] \\ &\quad + [e^{\sigma_\theta^2} - 2]r^2 \\ R^{[22]} &= \frac{1}{2}(r^2 + \sigma_r^2)[1 - e^{-2\sigma_\theta^2}] \\ R^{[12]} &= 0, \end{aligned} \quad (21)$$

whereby setting the azimuth to 0, there is no cross-correlation between the x-axis and y-axis, since the azimuth information has been used in the construction of $\mathbf{R}_{\hat{\mathbf{p}}}$. The covariance of the point is initially given by

$$\boldsymbol{\Sigma}_r = \begin{bmatrix} R^{[11]} & R^{[12]} \\ R^{[12]} & R^{[22]} \end{bmatrix}, \quad (22)$$

after which the tilt rotation is applied and the rotation $\mathbf{R}$ from world coordinates to radar coordinates is reversed

$$\boldsymbol{\Omega}_r = \mathbf{R}^T \mathbf{R}_{\hat{\mathbf{p}}}^T \begin{bmatrix} \boldsymbol{\Sigma}_r^{-1} & \mathbf{0} \\ \mathbf{0}^T & 0 \end{bmatrix} \mathbf{R}_{\hat{\mathbf{p}}} \mathbf{R}, \quad (23)$$

which yields a final Gaussian distribution of

$$\mathcal{N}(\mathbf{R}^T \hat{\mathbf{p}} + \mathbf{c}, (\boldsymbol{\Omega}_r)^{-1}). \quad (24)$$

Unlike the camera, combining $\mathbf{R}$ and $\mathbf{R}_{\hat{\mathbf{p}}}$ should not be performed to ensure that the orientation of $\boldsymbol{\Sigma}_r$ is kept intact. Figure 2 shows how the plane is tilted and how the covariance ellipse is tangential to the range-elevation arc.

This conversion method is based on the elevation of the point being a scalar value. If a more refined estimate of the elevation is known, such as from a Kalman Filter (KF), then making use of the full 3D UCM or the 3D decorrelated unbiased converted measurement (DUCM) should be used, which can also easily be brought into the world coordinate system.

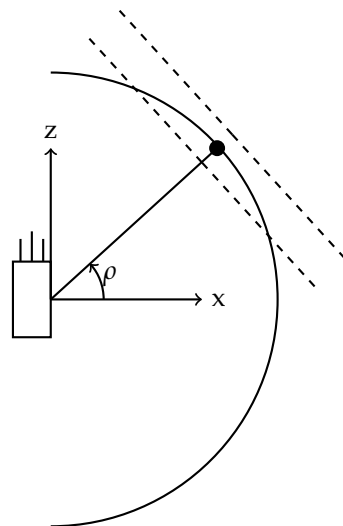


Figure 2. A side view of a radar and the tilted plane, at angle ρ , on which the rank-deficient covariance ellipse exists. The dashed lines show the one standard deviation of the ellipse at the POI for the radar sensor only.

3.5. Covariance Fusion

When fusing the precision matrices from all sensors, the cross-correlation between sensors is unknown. Thus, covariance intersection (CI) is required. To avoid iterative weight selection (which complicates differentiation through the loss), we use uniform CI weights, i.e., each sensor weight is set to the reciprocal of the number of sensors, $w_i = 1/S$.

Algorithm 1 provides the covariance-intersection fusion procedure. The algorithm takes as input the array of pixel values from each camera, the radars' range and azimuth measurements, the assumed λ and ρ , as well as the extrinsic parameters of all sensors. For each sensor, the projected Gaussian distributions are computed using (18) and (24). The information-form CI with uniform weights is

$$\Sigma \leftarrow \left(\frac{1}{S} \sum_{\Omega_i \in \Omega_s} \Omega_i \right)^{-1}, \quad (25)$$

where S is the number of sensors and Ω_s is the set of sensor precision matrices. The fused mean is obtained by applying the same weight to the information vector:

$$\mu \leftarrow \Sigma \left(\frac{1}{S} \sum_{\mu_i, \Omega_i \in \mu_s, \Omega_s} \Omega_i \mu_i \right), \quad (26)$$

where μ_s collects the individual sensor means. Equivalently, define the fused information matrix and vector as $\Omega = \frac{1}{S} \sum_i \Omega_i$ and $\eta = \frac{1}{S} \sum_i \Omega_i \mu_i$, then solve $\Omega \mu = \eta$ and set $\Sigma = \Omega^{-1}$. With uniform weights, the fused mean equals the standard information-fusion mean, while Σ is inflated relative to naive fusion, as desired under CI. The use of uniform weights is a simplification. If this algorithm is used merely for triangulation then the weights can be learnt through an iterative update approach. For the use in this application, having to update the weights introduces an optimization within an optimization; thus, it is left as uniform for the simplicity of the optimizer.

While the values of λ and ρ are not measured by the cameras and radars, respectively, they are required by Algorithm 1. As mentioned earlier, if this algorithm is used to perform covariance intersection, then λ and ρ can be iteratively estimated from an initial guess. Since Algorithm 1 is used as a step in sensor placement optimization, the true values of λ

and ρ are used. This ensures the precision matrices have the correct scale and orientation so that the intersected covariance accurately reflects the noise in the triangulated coordinates.

Algorithm 1 Covariance-intersection fusion

Input: Pixels $\mathbf{U}$, camera λ 's λ , radar measurements (with assumed elevations) $\mathbf{M}$, Cartesian camera centers $\mathbf{C}_c$, Cartesian radar centers $\mathbf{C}_r$, camera rotations $\mathbf{R}_c$, radar rotations $\mathbf{R}_r$

Output: Fused mean μ , covariance matrix Σ

```

1:  $\mu_s \leftarrow$  empty array
2:  $\Omega_s \leftarrow$  empty array
3: for all  $\{\mathbf{u}, \lambda, \mathbf{c}, \mathbf{R}\}$  in  $\{\mathbf{U}, \lambda, \mathbf{C}_c, \mathbf{R}_c\}$  do
4:   Acquire  $\hat{\mathbf{x}}$  using (14)
5:   Construct  $\mathbf{R}_{\hat{\mathbf{x}}}$  using (15)
6:   Compute  $\Omega_c$  using (17)
7:   Append  $\mathbf{c}$  to  $\mu_s$ 
8:   Append  $\Omega_c$  to  $\Omega_s$ 
9: end for
10: for all  $\{\mathbf{m}_r, \mathbf{c}, \mathbf{R}\}$  in  $\{\mathbf{M}, \mathbf{C}_r, \mathbf{R}_r\}$  do
11:   Acquire  $\hat{\mathbf{p}}$  using (19)
12:   Construct  $\mathbf{R}_{\hat{\mathbf{p}}}$  using (20)
13:   Compute  $\Omega_r$  using (23)
14:    $\mu_r \leftarrow \mathbf{R}^T \hat{\mathbf{p}} + \mathbf{c}$ 
15:   Append  $\mu_r$  to  $\mu_s$ 
16:   Append  $\Omega_r$  to  $\Omega_s$ 
17: end for
18: Compute  $\Sigma$  using (25)
19: Compute  $\mu$  using (26)
20: return  $\mu, \Sigma$ 

```

It should be noted that the intersected covariance matrix can be rank-deficient when the zero-precision axes of multiple sensors align (e.g., cameras viewing along a common line through the POI, or radars whose zero-precision axes lie in a common plane).

4. Gradient Descent Approach and Constraints

4.1. Constraints and Considerations

While the sensor centers are needed in Cartesian coordinates to perform a linear mapping of a POI onto the sensor plane, applying GD to Cartesian centers will result in undesired behavior, owing to how uncertainty scales with distance. To illustrate this point, the Cartesian size of the covariance ellipsoid for the cameras is proportional to the λ , which itself is proportional to the distance that the point is from the camera center. Thus, points closer to the camera will have a lower covariance ellipsoid of the image plane than points further away. This makes intuitive sense. Optimizing the camera's centers in Cartesian coordinates will result in the cameras moving closer to the POI coordinate mean in order to minimize measurement uncertainty. This is not practically achievable as cameras need to be a set distance from the ROI in order to keep the ROI within the FOV of the cameras.

Thus the sensor centers must be constrained. A useful approach will be to have the position of the sensors depend on some angle from the center of the ROI, which will be the main parameter on which GD is applied. This would imply moving the sensors along a predefined trajectory or parametric function such that the sensors' FOVs remain centered on the center of the ROI. This implies that the sensors' rotations need to be adjusted accordingly.

This makes the sensor rotation matrix a function of the sensor center. The rotation matrix can be constructed by pointing one of the axes, defined as the a -axis, towards the mean of the POIs, $\bar{\mathbf{x}}$.

$$\mathbf{e}'_a = \frac{\bar{\mathbf{x}}}{\|\bar{\mathbf{x}}\|}.$$

The second axis can be chosen to be parallel to the world coordinate xy -plane,

$$\mathbf{e}'_b = \frac{\begin{bmatrix} -(\mathbf{e}'_a)_y & (\mathbf{e}'_a)_x & 0 \end{bmatrix}^T}{\| \begin{bmatrix} -(\mathbf{e}'_a)_y & (\mathbf{e}'_a)_x & 0 \end{bmatrix}^T \|},$$

where $(\cdot)_x$ and $(\cdot)_y$ are the x and y values of $(\cdot)$. The final axis is then obtained using the cross-product

$$\mathbf{e}'_c = \mathbf{e}'_a \times \mathbf{e}'_b,$$

and the rotation matrix is constructed,

$$\mathbf{R}_s^{(a)} = \begin{bmatrix} (\mathbf{e}'_a)^T \\ (\mathbf{e}'_b)^T \\ (\mathbf{e}'_c)^T \end{bmatrix}, \quad (27)$$

where the a -axis is aligned with the z -axis for the camera, and the x -axis for the radar, and the b -axis is aligned with the x -axis and the y -axis for the camera and radar respectively. This will point the sensors towards the mean of POIs in the ROI, using only a pitch and a yaw on the orientation of the sensors. An example is given in Figure 3, where a sensor's center is set at a specific range and height, while being allowed to vary in azimuth.

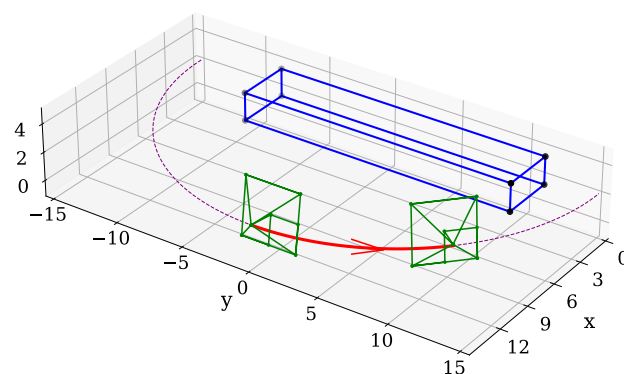


Figure 3. The blue rectangle represents the ROI, the purple dotted line is the path that the sensors are allowed to move on, the red arrow shows the movement of the sensor in one iteration step, and the green shows the centre and image plane of the sensor. Note that the orientation of the sensor changes in only pitch and yaw. The horizontal lines in the sensor's plane, show two of the three axes of the sensor.

One point of concern is the inversion of the sum of precision matrices to obtain Σ in Algorithm 1. If the condition number of this matrix is large, it will be harder to invert. This can be problematic, but it is unavoidable. Another point of consideration is that different noise factors between the sensors will result in different step sizes within the GD algorithm, as the level of contribution to the final covariance will depend on the noise parameters of the sensors. A method to circumvent this is discussed later.

4.2. Gradient Descent

The choice to use GD as the optimization algorithm is to showcase the ability of the CI-triangulation algorithm to have a useable derivative. Any optimization scheme can be used here to find the optimal placement of the sensors.

The loss function for the GD algorithm is given in Algorithm 2. It takes as arguments the POIs, and the camera and radar centers that can exist in any coordinate space. Since Algorithm 1 requires Cartesian sensor centers, the camera and radar centers are converted to xyz coordinates. These are then used to construct the rotation matrices for the sensors, which completes the extrinsic parameters of the sensors. The POIs are then projected onto each sensor using (3), (6) and (7). Furthermore, the λ and ρ for each of the POIs are also saved. In the initial pass, the mean values of λ and ρ for each camera and radar are used as an estimate, these can also be chosen based on the users knowledge of the environment. This allows Algorithm 1 to obtain a triangulated point, from which an updated estimate of λ and ρ are found for each POI. These estimates is then passed to Algorithm 1 for a second time to acquire a finer estimate of the covariance matrix, from which the loss for the POIs is calculated, using one of the metrics from Table 1, and averaged. During tests, the two-step algorithm with an initial mean λ and ρ was always able to match the triangulated point of the CI-triangulation where the true values of λ and ρ were used. Further testing showed that the λ can be arbitrarily set, so long as the initial values of λ were the same for all the cameras, as it only scales the camera covariance without changing the mean line. For ρ , an initial value that is within the expected range of elevation is required for a two-step approach; if not, multiple iterations are required for convergence to occur.

Algorithm 2 Covariance fusion loss function with iterative λ and ρ estimation

Input: 3D points $\mathbf{X}$, camera centers $\mathbf{C}_c$, Radar centers $\mathbf{C}_r$, mean camera depth $\bar{\lambda}$, mean radar elevation $\bar{\rho}$

Output: Loss l

```

1:  $\mathbf{C}'_c \leftarrow$  conversion of  $\mathbf{C}_c$  to Cartesian coordinates
2:  $\mathbf{C}'_r \leftarrow$  conversion of  $\mathbf{C}_r$  to Cartesian coordinates
3:  $\mathbf{R}_c \leftarrow$  alignment rotation matrices for cameras using (27), with  $\bar{\mathbf{x}}$  as the mean environment point
4:  $\mathbf{R}_r \leftarrow$  alignment rotation matrices for radars using (27), with  $\bar{\mathbf{x}}$  as the mean environment point
5:  $\mathbf{I} \leftarrow$  empty array
6: for all  $\mathbf{x} \in \mathbf{X}$  do
7:    $\mathbf{U} \leftarrow$  empty array {Camera projections}
8:    $\mathbf{\Lambda} \leftarrow$  empty array {Camera depths}
9:   for all camera in cameras do
10:    Project  $\mathbf{x}$  to camera: obtain image point  $\mathbf{u}$  (3)
11:    Append  $\mathbf{u}$  to  $\mathbf{U}$ 
12:   end for
13:    $\mathbf{M} \leftarrow$  empty array {Radar projections}
14:    $\mathbf{P}_\rho \leftarrow$  empty array {Radar elevations}
15:   for all radar in radars do
16:    Project  $\mathbf{x}$  to radar: obtain  $[r, \theta]^T$  using (6)-(8)
17:    Append  $[r, \theta, \bar{\rho}]^T$  to  $\mathbf{M}$ 
18:   end for
19:    $(\hat{\mu}, \hat{\Sigma}) \leftarrow$  covariance fusion of  $\mathbf{U}$ , array of  $\bar{\lambda}$ ,  $\mathbf{M}$ ,  $\mathbf{C}'_c$ ,  $\mathbf{C}'_r$ ,  $\mathbf{R}_c$ ,  $\mathbf{R}_r$  with these values
20:   For each camera, re-project  $\hat{\mu}$  to get estimated  $\hat{\lambda}$ 
21:   For each radar, re-project  $\hat{\mu}$  to get estimated  $\hat{\rho}$ 
22:   Replace  $\bar{\rho}$  with  $\hat{\rho}$  in  $\mathbf{M}$ 
23:    $(\hat{\mu}, \hat{\Sigma}) \leftarrow$  covariance fusion of  $\mathbf{U}$ ,  $\hat{\lambda}$ ,  $\mathbf{M}$ ,  $\mathbf{C}'_c$ ,  $\mathbf{C}'_r$ ,  $\mathbf{R}_c$ ,  $\mathbf{R}_r$  with estimated  $\lambda$  and  $\rho$ 
24:    $l_x \leftarrow \text{lossFunction}(\hat{\Sigma})$ 
25:   Append  $l_x$  to  $\mathbf{I}$ 
26: end for
27:  $l \leftarrow \text{mean}(\mathbf{I})$ 
28: return  $l$ 

```

The loss function in Algorithm 2 can vary, with different functions minimizing different criteria. Table 1 provides a list of some potential loss functions. These are functions that convert a matrix to a scalar value, namely, the determinant, trace, and RMSE of a 3×3 matrix. Geometrically, the determinant of a covariance matrix is a representation of the volume of the covariance ellipsoid, while the trace is the total variation of the distribution, disregarding the correlation between variables. The RMSE is a measure of the error of the triangulated coordinate.

Because of the nature of the loss function and the fusion algorithm, it is recommended to use an instantiation of one of the tools in the class of *auto-diff* algorithms to obtain an estimate of the derivative of the loss function.

Table 1. Matrix norms and their geometric meaning. The loss is a function of one of these matrix norms applied to the covariance matrix of a point Σ .

Optimality	Loss Function	Geometric Meaning
D-optimality	$\ln \det(\Sigma)$	A measure of the “volume” of the covariance matrix
A-optimality	$\ln \text{trace}(\Sigma)$	Natural log of the sum of the eigenvalues of the covariance matrix.
ine -	$\sqrt{\frac{1}{3} \text{trace}(\Sigma)}$	RMSE of the covariance function

5. Results

The simulation of the sensor placement optimization is challenging owing to the large number of initial conditions, which include the sensor positions, the sensor noise values, and the number and type of sensors used. This is further exacerbated by a large set of POIs within the ROI for which the sensors must be optimized.

The algorithm is tested on a cricket ball tracking use case, which is typically used for assisted umpiring and match analysis in sports broadcasting. The ball is modelled using equations of motion, including drag and lift forces, as well as spin. The spin is modelled using the equations from [29,30]. When the ball touches the ground, the velocity of the z-axis is inverted and multiplied by 0.5, which is meant to represent the energy lost during the bounce of the ball, called the coefficient of restitution. Figure 4 shows 200 simulated trajectories, which will be used as the set of POIs (135 POIs per delivery), where Δt was set to 1/180 for a max simulation time of 0.75 s. The initial positions and velocities of the balls were set using Gaussian distributions,

$$\mathbf{x}_0 = \mathcal{N}\left(\begin{bmatrix} 0 \\ 0 \\ 2.0 \end{bmatrix}, \begin{bmatrix} 0.5 & 0 & 0 \\ 0 & 0.1 & 0 \\ 0 & 0 & 0.01 \end{bmatrix}\right),$$

$$\mathbf{v}_0 = \mathcal{N}\left(\begin{bmatrix} 0 \\ 27 \\ -3.8 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.1 \end{bmatrix}\right),$$

to introduce randomness into the paths of the balls. The positions of the balls are offset so that their mean is at the origin of the world coordinates. This makes positioning and constraining the sensors easier.

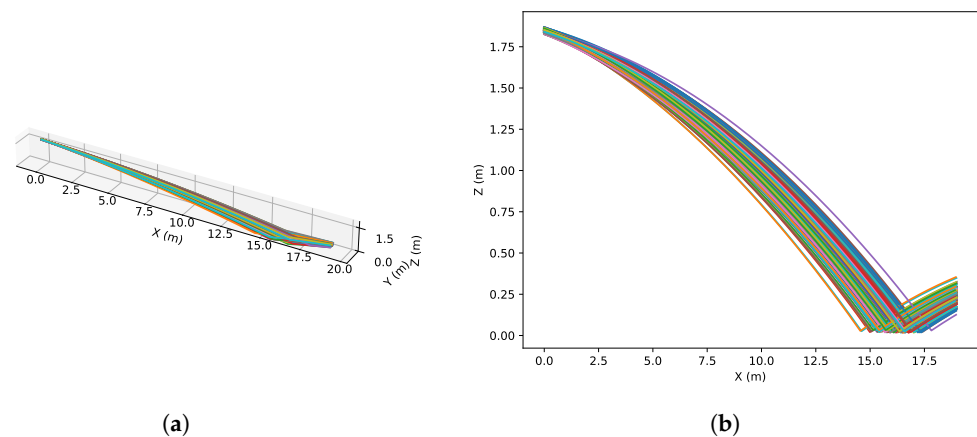


Figure 4. Paths of the simulated balls, where each colored line is a simulated ball. (a) shows the 3D paths of the balls, while (b) shows the side view of the balls.

The sensors are positioned using cylindrical coordinates, where only the azimuth of the sensors is changed using GD. The range and height of the cameras are set to 25 and 10 m, respectively, while the radar range and height are set to 15 and 0 m.

As noted earlier, the range of the camera can change and still yield the same image, so long as the focal length of the camera also changes proportionally. To ensure that the ball will always be in the camera image, the intrinsic parameters of the camera are set to

$$\mathbf{K} = \begin{bmatrix} 800 & 0 & 960 \\ 0 & 3600 & 540 \\ 0 & 0 & 1 \end{bmatrix}.$$

For the radar, the max range is set to 30 m with a max azimuth angle of 60° . This, too, is set so that the ball will always be viewed by the radar.

For the environment, it is assumed that there are no occlusions by the players; occlusions are uncommon in cricket broadcasting. Furthermore, it is assumed that the cameras are placed on the roof of the stadium and that the radars can be placed on the ground.

It was found that if a single optimizer was used on different types of sensors, those with a large variance in a certain dimension failed to be optimized, and the optimizer focused its efforts on the accurate sensor types in that dimension. Consequently, each sensor position is individually optimized according to the covariance matrix of its sensor type. This does present complications since the surface of the loss function will change per iteration step, as one group of sensors acts independently from the other and changes the surface of the loss function in an unexpected way. This complication can, however, be mitigated by increasing the number of optimization iteration steps and by updating only one sensor type per iteration step. The use of different optimizers ensures that each sensor type is properly placed while lowering the effects that the sensor noises have on the movement of the sensors.

The chosen optimizer is the ADAM optimizer [31], which will adapt the learning rates for each sensor type, mitigating the complication of different sensors having different noise values. ADAM is initialized with $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1 \times 10^{-8}$, and with a learning rate of $\alpha = 0.02$ for all sensor types. The use of ADAM also removes the need to change the optimizer when changing the minimization criteria, as different minimization criteria will require different learning rates to provide optimal results if stochastic GD (SGD) is used.

The noise of the camera is chosen to be a zero-mean Gaussian distribution with a standard deviation of three pixels. This is meant to encompass the noise in the projection

process and the noise in detecting the ball within the image. The radar range and azimuth noise are set as a zero-mean Gaussian distribution with a standard deviation of 0.01 meters and 0.15 rad respectively, which is indicative of a mm-wave radar. These are noise values are indicative of what has been experienced using a mm-wave radar and cameras in cricket ball tracking.

To test the optimization scheme, a Monte Carlo simulation is performed where the initial orientation of the sensors is changed for each trial. A total number of 4500 trials is performed, and the number of iteration steps is set to 500, which, in testing the system, is long enough for the different minimization criteria to stabilize. To initialize the sensor types, each sensor is given a random angle $\theta \in [0, 2\pi)$ such that no two sensors in the same sensor type share the same angle. This is performed for different sensor configurations, where the number of cameras and radars is varied. The results are shown in Table 2, where, if the sensors are positioned such that the CI-triangulation is unable to triangulate a point, the run is excluded. This is predominately for radar-only setups, due to the z-axis having a large variance value, and the triangulation struggling due to an estimate of ρ being used initially. For radar-camera, or purely camera, setups, above 90% of the runs are used, but for purely radar setups, the percentage drops to 80% for three and four radars, and to 40% for two-radar setups. This shows the utility of cameras in such a system, as one camera and one radar are able to easily triangulate a point. An example of the sensor movement is shown in Figure 5 for when the determinant is the metric in the loss function.

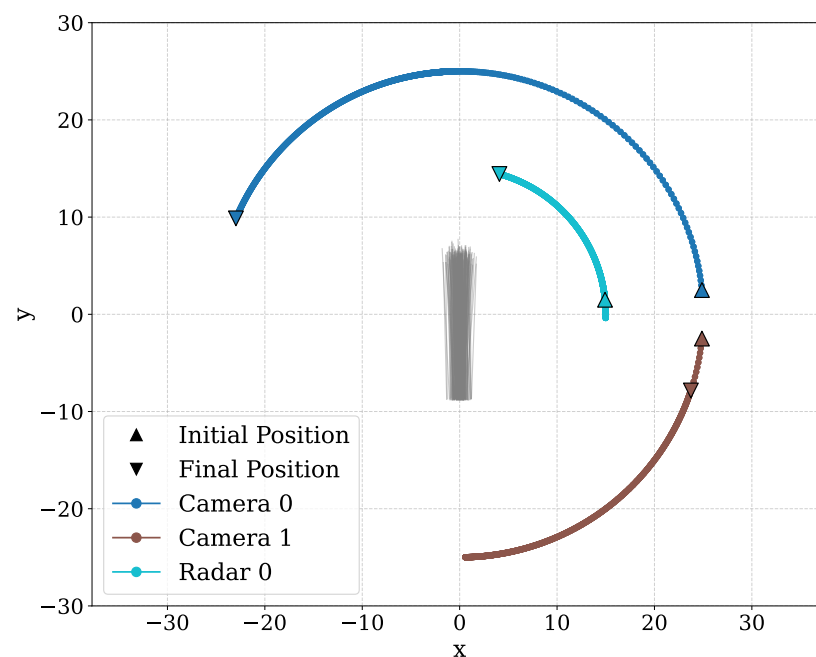


Figure 5. A BEV of the movement of the sensors for a one radar, two camera system, where the grey is the simulated balls. The determinant is the loss function utilized in this setup. Each dot represents a GD iteration step. Interestingly, the camera moves back onto itself, which is caused by the changing surface of the loss function at each second iteration.

Table 2. Summary of log-normal mean RMSE and its standard deviation at the beginning and end of optimization for each sensor configuration and minimization criterion for the 4500 Monto Carlo runs. The start mean and standard deviation are from the random initial placements of the sensors, while the end mean and standard deviation are after the sensor placement optimization. The table also reports the absolute and percentage decrease in RMSE. Bold numbers show the best performing minimization criteria and best performing sensor configuration for the sensor configuration group.

Sensor Configuration		Start		Minimization	End		$\Delta\mu$ (mm)	% Decrease
No. Cameras	No. Radars	μ (mm)	σ (mm)	Criteria	μ (mm)	σ (mm)		
4	0	78.180	15.10	det	70.966	8.28	−7.215	9.228
				trc	71.070	1.95	−7.110	9.094
				RMSE	71.006	2.02	−7.174	9.177
3	1	78.249	18.58	det	68.426	10.27	−9.824	12.555
				trc	61.041	7.75	−17.209	21.992
				RMSE	59.867	7.06	− 18.383	23.493
2	2	81.257	19.94	det	59.027	10.75	− 22.231	27.358
				trc	62.020	10.28	−19.237	23.675
				RMSE	62.087	10.87	−19.170	23.592
1	3	90.309	18.22	det	69.291	0.99	−21.018	23.274
				trc	68.671	0.75	−21.638	23.960
				RMSE	68.571	0.76	− 21.738	24.071
0	4	1095.817	1992.16	det	160.197	99.95	− 935.619	85.381
				trc	161.310	102.00	−934.507	85.279
				RMSE	173.491	96.32	−922.325	84.168
3	0	90.608	28.94	det	77.381	8.64	−13.227	14.598
				trc	71.499	0.65	−19.109	21.090
				RMSE	71.482	1.21	− 19.126	21.108
2	1	96.442	35.12	det	63.003	13.74	−33.439	34.673
				trc	60.616	11.56	−35.826	37.148
				RMSE	60.336	11.29	− 36.106	37.438
1	2	111.013	35.56	det	73.077	1.37	−37.936	34.173
				trc	72.612	1.86	−38.401	34.591
				RMSE	72.520	1.94	− 38.494	34.675
0	3	2437.991	5890.00	det	223.184	192.45	−2214.808	90.846
				trc	235.126	214.61	−2202.866	90.356
				RMSE	217.745	152.83	− 2220.246	91.069
2	0	166.338	145.69	det	81.468	1.97	−84.870	51.022
				trc	70.511	2.18	− 95.827	57.610
				RMSE	72.179	7.74	−94.159	56.607
1	1	201.595	109.51	det	95.421	2.71	− 106.173	52.667
				trc	97.579	4.75	−104.016	51.597
				RMSE	98.460	10.21	−103.134	51.159
0	2	6135.995	7567.36	det	494.726	206.24	− 5641.269	91.937
				trc	1198.169	1464.78	−4937.826	80.473
				RMSE	771.072	231.43	−5364.923	87.434
4	4	64.521	7.38	det	60.953	3.50	−3.568	5.530
				trc	60.490	5.25	−4.031	6.248
				RMSE	60.083	5.54	− 4.438	6.879
3	3	68.766	10.28	det	62.694	4.49	−6.072	8.830
				trc	62.674	6.54	−6.092	8.859
				RMSE	61.922	6.92	− 6.844	9.952

6. Discussion

The final RMSE values show that, as expected, as the number of sensors used increases, the RMSE decreases. Furthermore, within each subgroup, i.e., the four-total, three-total, and two-total subgroups, there is a definite configuration that performs the best, this being the (2-2), (2-1), and (2-0) configurations, where (c-r) are the number of cameras and radars respectively, which show a RMSE decrease of 22.231, 36.106, and 95.827 respectively. For combinations with a low number of sensor, the percentage decrease is dramatic. This is expected, as greater care must be taken when placing few sensors to ensure that the placement allows the sensors to work with their strengths. This is most notable in the two-total subgroup.

An interesting note is that, in purely radar systems, a large percentage decrease occurs. This implies that the placement of the radars purely radar systems is vital for performance and decreasing the RMSE. The initial random placement for the radars produces an RMSE with a large mean and standard deviation. This could occur if the radars are placed close together, making their azimuth measurements too similar which can bias the mean estimate towards the radars. After the optimization is performed, the radars still produce the highest RMSE; further investigation is required to discern why this is the case. This can also be seen in Figure 6, where there are distinct positions that the radars tend towards, most notably pointing down/up the pitch. The cameras have a similar placement in Figure 6, but have a wider variance over the placement of the cameras.

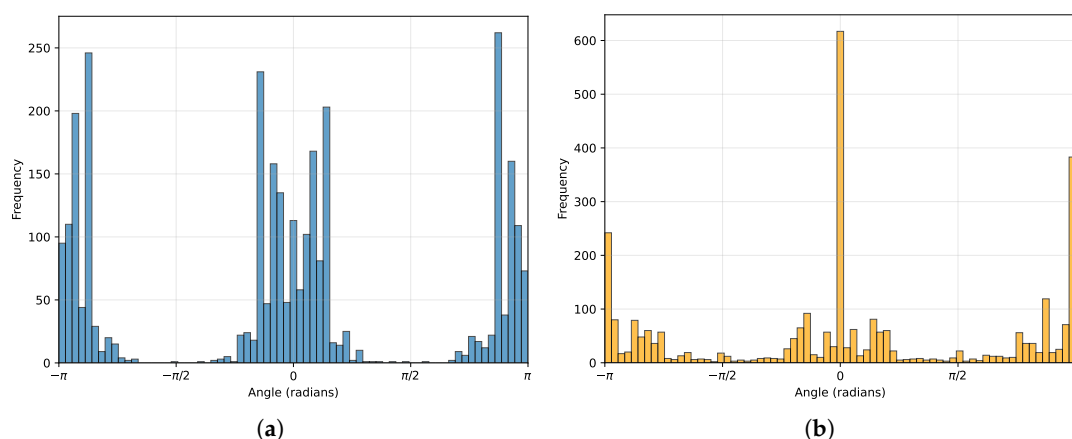


Figure 6. Histogram plots of the final azimuth angle (in radians) of the sensors in a (2-2) configuration, (a) is the camera's final azimuth, and (b) is for the radar's final azimuth. The camera's histogram shows that there are broader groups that the cameras tend towards. This is not the case for the radars, where there is a greater, single azimuth angle that the radars move towards. This could be due to the radar measuring the balls' azimuth, and thus need to be in a position where this measurement can be fully utilized. An interesting note is that the cameras and radars tend towards the same azimuth points, which are on either side of the length of the pitch.

Table 2 shows that there is no RMSE loss minimization metric that consistently outperforms the others. This is interesting as even though the determinant captures more of the covariance matrix, it does not produce results that outperform the trace or RMSE metrics. This makes sense given the fact that it is the RMSE that is being minimized, which only depends on the diagonal of the covariance matrix. Even though this is the case, working towards a smaller covariance ellipsoid, through the determinant, was expected to produce better results.

The results for the same number of radars and cameras show that after (2-2), there are diminishing returns, with a (2-2) configuration performing the same as the (3-3) and (4-4) configurations. However, with the (3-3) and (4-4) configurations, there is a lower

RMSE standard deviation, which implies that the performance of these configurations is more consistent. This is intuitive, as the more sensors there are, the more constrained the covariance matrix is, where, eventually, the position of the sensors is redundant. This also explains why there is a small decrease in RMSE when there are more sensors present: the random initial starting positions for the sensors have a higher chance of already highly constraining the covariance of the points. In terms of practicality, having a (2-2) system offers the best performance for a minimal number of sensors, provided the sensors are optimally placed.

For most of the sensor configurations, the final RMSE standard deviations are low. This implies that there is a globally optimal configuration for the sensors, which the algorithm finds.

7. Conclusions

The algorithm is tested on a cricket ball tracking use case, which is typically used for assisted umpiring and match analysis in sports broadcasting. The results show that there are specific combinations and placements that offer superior performance if the sensors are placed correctly and that as the number of sensors increases, there are diminishing returns on the change of the RMSE, while finding placements of the sensors that minimize the RMSE for that specific sensor configuration.

In future work, the Doppler velocity of the radar can be incorporated into the optimization scheme. Furthermore, if specific regions require a stronger triangulation, such as the area before the cricket ball bounces and at the wickets, a weighting scheme can be used to force the sensors to place a higher priority on these regions.

Author Contributions: Conceptualization, D.K. and J.P.d.V.; methodology, D.K.; software, D.K.; validation, D.K.; formal analysis, D.K.; investigation, D.K.; resources, D.K. and J.P.d.V.; data curation, D.K.; writing—original draft preparation, D.K.; writing—review and editing, J.P.d.V.; visualization, D.K.; supervision, J.P.d.V.; project administration, D.K.; funding acquisition, J.P.d.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the MultiChoice Chair of Machine Learning.

Institutional Review Board Statement: The study was approved by the EBIT Faculty Committee for Research Ethics & Integrity of the University of Pretoria (protocol code EBIT/3/2025 approved on 4 May 2024).

Informed Consent Statement: Not applicable.

Data Availability Statement: The ball trajectories are stored in *ball_state_spaces.npy*. The shape of the trajectories is (state space, ball number, time). The state space is of dimension 12, where the first three dimensions are for position (x, y, z) (m), then velocity, acceleration, and angular velocity.

Acknowledgments: During the preparation of this manuscript/study, the authors used Grammarly v1.156.1.0 for the purposes of language editing. The authors have reviewed and edited the output and take full responsibility for the content of this publication.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Ahn, J.W.; Chang, T.W.; Lee, S.H.; Seo, Y.W. Two-Phase Algorithm for Optimal Camera Placement. *Sci. Program.* **2016**, *2016*, 4801784. [CrossRef]
2. Hörster, E.; Lienhart, R. Optimal Placement of Multiple Visual Sensors. In *Multi-Camera Networks*; Elsevier: Amsterdam, The Netherlands, 2009; pp. 117–138. [CrossRef]
3. Gungor, O.; Rosing, T.S.; Aksanli, B. RESPIRE++: Robust Indoor Sensor Placement Optimization Under Distance Uncertainty. *IEEE Sens. J.* **2022**, *22*, 11355–11363. [CrossRef]

4. Vlasenko, I.; Nikolaidis, I.; Stroulia, E. The Smart-Condo: Optimizing Sensor Placement for Indoor Localization. *IEEE Trans. Syst. Man Cybern. Syst.* **2015**, *45*, 436–453. [\[CrossRef\]](#)
5. Rahimian, P.; Kearney, J.K. Optimal Camera Placement for Motion Capture Systems. *IEEE Trans. Vis. Comput. Graph.* **2017**, *23*, 1209–1221. [\[CrossRef\]](#)
6. Kwon, B.; Huh, J.; Lee, K.; Lee, S. Optimal Camera Point Selection Toward the Most Preferable View of 3-D Human Pose. *IEEE Trans. Syst. Man Cybern. Syst.* **2022**, *52*, 533–553. [\[CrossRef\]](#)
7. Deng, H.; Zhou, J.; Zhao, Z.; Zhang, L. Sensor Placement Optimization and Rapid Deployment of a Class of Visual Sensor Networks. In Proceedings of the 2023 42nd Chinese Control Conference (CCC), Tianjin, China, 24–26 July 2023; pp. 6215–6220. [\[CrossRef\]](#)
8. Joshi, S.; Boyd, S. Sensor Selection via Convex Optimization. *IEEE Trans. Signal Process.* **2009**, *57*, 451–462. [\[CrossRef\]](#)
9. Ye, L.; Roy, S.; Sundaram, S. Resilient Sensor Placement for Kalman Filtering in Networked Systems: Complexity and Algorithms. *IEEE Trans. Control Netw. Syst.* **2020**, *7*, 1870–1881. [\[CrossRef\]](#)
10. Tzoumas, V.; Jadbabaie, A.; Pappas, G.J. Sensor placement for optimal Kalman filtering: Fundamental limits, submodularity, and algorithms. In Proceedings of the 2016 American Control Conference (ACC), Boston, MA, USA, 6–8 July 2016; pp. 191–196. [\[CrossRef\]](#)
11. Akbarzadeh, V.; Gagne, C.; Parizeau, M.; Argany, M.; Mostafavi, M.A. Probabilistic Sensing Model for Sensor Placement Optimization Based on Line-of-Sight Coverage. *IEEE Trans. Instrum. Meas.* **2013**, *62*, 293–303. [\[CrossRef\]](#)
12. Nguyen, T.T.; Thanh, H.D.; Hoang Son, L.; Trong Le, V. Optimization for the sensor placement problem in 3D environments. In Proceedings of the 2015 IEEE 12th International Conference on Networking, Sensing and Control, Taipei, Taiwan, 9–11 April 2015; pp. 327–333. [\[CrossRef\]](#)
13. Suresh, M.S.S.; Narayanan, A.; Menon, V. Maximizing Camera Coverage in Multicamera Surveillance Networks. *IEEE Sens. J.* **2020**, *20*, 10170–10178. [\[CrossRef\]](#)
14. Krause, A.; Singh, A.; Guestrin, C. Near-Optimal Sensor Placements in Gaussian Processes: Theory, Efficient Algorithms and Empirical Studies. *J. Mach. Learn. Res.* **2008**, *9*, 235–284.
15. Jiang, C.; Soh, Y.C.; Li, H. Sensor Placement by Maximal Projection on Minimum Eigenspace for Linear Inverse Problems. *IEEE Trans. Signal Process.* **2016**, *64*, 5595–5610. [\[CrossRef\]](#)
16. Haralick, R. Propagating covariance in computer vision. In Proceedings of the 12th International Conference on Pattern Recognition, Jerusalem, Israel, 9–13 October 1994; Volume 1, pp. 493–498. [\[CrossRef\]](#)
17. Muratoglu, A.; Söken, H.E.; Soergel, U. Error Covariance Analyses for Celestial Triangulation and Its Optimality: Improved Linear Optimal Sine Triangulation. *Aerospace* **2025**, *12*, 385. [\[CrossRef\]](#)
18. Hedborg, J.; Robinson, A.; Felsberg, M. Robust Three-View Triangulation Done Fast. In Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops, Columbus, OH, USA, 23–28 June 2014. [\[CrossRef\]](#)
19. Kukelova, Z.; Pajdla, T.; Bujnak, M. Fast and Stable Algebraic Solution to L2 Three-View Triangulation. In Proceedings of the 2013 International Conference on 3D Vision, Seattle, WA, USA, 29 June–1 July 2013. [\[CrossRef\]](#)
20. Byröd, M.; Josephson, K.; Åström, K. Fast Optimal Three View Triangulation. In *Computer Vision—ACCV 2007*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 549–559. [\[CrossRef\]](#)
21. Liggins, M.; Hall, D.; Llinas, J. *Handbook of Multisensor Data Fusion*; CRC Press: Boca Raton, FL, USA, 2017. [\[CrossRef\]](#)
22. Yuan, J.; Zhu, S.; Tang, K.; Sun, Q. ORB-TEDM: An RGB-D SLAM Approach Fusing ORB Triangulation Estimates and Depth Measurements. *IEEE Trans. Instrum. Meas.* **2022**, *71*, 5006315. [\[CrossRef\]](#)
23. Kim, H.; Lee, H.; Lee, S. A new covariance intersection based integrated SLAM framework for 3D outdoor agricultural applications. *Electron. Lett.* **2024**, *60*, e13206. [\[CrossRef\]](#)
24. Chen, S.; Frémont, V. A loosely coupled vision-LiDAR odometry using covariance intersection filtering. In Proceedings of the 2021 IEEE Intelligent Vehicles Symposium (IV), Nagoya, Japan, 11–17 July 2021; pp. 1102–1107.
25. Hartley, R.I.; Sturm, P. Triangulation. *Comput. Vis. Image Underst.* **1997**, *68*, 146–157. [\[CrossRef\]](#)
26. Houssineau, J.; Clark, D.E.; Ivekovic, S.; Lee, C.S.; Franco, J. A Unified Approach for Multi-Object Triangulation, Tracking and Camera Calibration. *IEEE Trans. Signal Process.* **2016**, *64*, 2934–2948. [\[CrossRef\]](#)
27. Houssineau, J.; Ivekovic, S.; Clark, D.E. Disparity space: A parameterisation for Bayesian triangulation from multiple cameras. In Proceedings of the 2012 15th International Conference on Information Fusion, Singapore, 9–12 July 2012; pp. 1734–1740.
28. Longbin, M.; Xiaoquan, S.; Yiyu, Z.; Zhongkang, S. Unbiased converted measurements for target tracking. In Proceedings of the IEEE 1997 National Aerospace and Electronics Conference. NAECON 1997, Dayton, OH, USA, 14–17 July 1997; Volume 2, pp. 1039–1041. [\[CrossRef\]](#)
29. Cross, R. Bounce of a spinning ball near normal incidence. *Am. J. Phys.* **2005**, *73*, 914–920. [\[CrossRef\]](#)

30. Grimshaw, S.D.; Briggs, A.; Atkins, N.R. A review and reassessment of the aerodynamics of cricket ball swing. *Flow* **2024**, *4*, E6. [[CrossRef](#)]
31. Kingma, D.P.; Ba, J. Adam: A Method for Stochastic Optimization. *arXiv* **2017**, arXiv:1412.6980. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.