IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received March 6, 2022, accepted May 20, 2022, date of publication May 30, 2022, date of current version June 6, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3178832

Evolution Based Single Camera Resectioning Based on Distance Maps of a Known Geometry for Squash Sports

C. BRUMANN AND M. KUKUK
Department of Computer Science, University of Applied Sciences and Arts Dortmund, 44139 Dortmund, Germany
Corresponding author: C. Brumann (christopher.brumann@fh-dortmund.de)

ABSTRACT Nowadays, video recordings of sport events is standard practice for a variety of applications, ranging from entertainment to competition analysis. Beside that, analysis of athletes while exercising is of particular interest for their coaches in order to gain insight into training quality and allow for training control. To bring together video recordings and the desire for analysis we present the implementation of a genetic algorithm (GA) for the important step of camera calibration. Our implementation can be used not only in a prospective but also in a retrospective manner for the squash sport. We do not rely on directly or manually provided image-world coordinates, but rather only on the playing field as known geometric object, present in the physical camera's captured image. To find the best GA configuration, we evaluate all combinations of 2 initialization-, 2 fitness-, 3 selection-, 4 crossover-, and 2 mutation strategies. We apply and evaluate the GA's accuracy on synthetic, artificial renderings, and real world data as well as comparing it to other standard optimization algorithms. Our results reveal the importance of correct camera placement and show sufficient accuracy for our goal of athlete movement analysis. The results will serve for a automatic athlete movement analysis tool to support squash specific training procedures.

INDEX TERMS Camera calibration, genetic algorithm, position estimation, squash sport, videos.

I. INTRODUCTION

Camera resectioning, also known as geometric camera calibration, is typically the first step in many computer vision applications [1]–[4]. It is known as the process of estimating intrinsic and extrinsic camera parameters. While intrinsic parameters define the camera's focal length and principal point, extrinsic parameters denote the camera pose as rotation and translation transformation from world to camera coordinates. Thus, in combination, they reflect the relationship between the captured world scene and the images acquired by the camera. In general, stereo camera calibration is required in applications, where cameras are used for obtaining absolute quantitative measurements, such as position, distance, velocity, size, or angle with respect to a world coordinate system. Applications for that can be found in the medical domain for X-Ray calibration, where an attached color camera [3] with additional depth information (RGB-D) [4] is used. However, in contrast to using time-of-flight sensors or a multiple view camera setup, depth information is lost when using a single stationary camera. On the other hand, if such cameras are calibrated, quantitative measurements on planar objects can still be obtained. Applications are calibration of action cameras for photogrammetric purposes [5] or using roadside traffic management cameras for vehicle speed estimation [6]. Further, for educational purposes, camera resectioning has been used for visual tracking in Aikido [7]. Here we present a single camera calibration method, developed for the specific requirements posed by sports applications. Specifically, we are interested in the game of squash and aim for providing a tool for coaches and athletes to optimize the training process and improve performance during competition. This is of particular interest, since well-planned and executed training not only leads to physical adaptation mechanisms, but also reduces the risk for injury [8].

In general squash is an agile racket sport played on a squash court by two athletes on a $6.4\,\text{m} \times 9.75\,\text{m}$ shared playing field, which is surrounded by four walls and has been characterized as a complex sport which demands a combination of physical, technical, and tactical abilities [9]. Lines relevant to the game are well defined [10] and are present on the two side walls, the front wall as well as on the floor. For broadcasts

The associate editor coordinating the review of this manuscript and approving it for publication was Varuna De Silva.

and video recordings of squash matches the camera position corresponds to a typical viewer position centered behind the court. After initial setup, camera parameters do not change during matchplay, resulting in a stationary view. This perspective represents the preferred view for broadcasting large tournaments, since it can easily be setup to capture the entire court.

During a rally, both athletes play the ball alternating as to hit the front wall directly or indirectly (via a wall). One special characteristic of squash is that both players share the same physical space, while they must not interfere with each other. Thus intelligent shot selection, body positioning, and motion patterns are important elements in the game of squash. In [11] dynamic motions for players with different technical abilities have been studied and it was reported that athletes' position was closer to the front wall for winning rallies across different skill levels. More recent research investigated the difference in work-rate as a function of game duration and distance covered between international and Slovene national players, founding that international players cover larger distances caused by longer average game duration [12]. In addition, the importance of a specific area on the court floor, called the T area was investigated [13], revealing that controlling this area relates to player's dominance during a rally and ultimately with the likeliness of winning that rally. Further, awareness the opponent's motion is essential to retaining control of a rally. Moreover, it is significant that squash requires quick reactions to opponent's actions. Thus there remains only a short time for making tactical decisions, and as a consequence, internalizing certain patterns during training and retrieving them in competition is an important skill. A recent study found six situation awareness cluster including players' position on court and found differences in expert athletes' behavior [14]. This empathizes the importance of this topic even for elite squash athletes. Further, individual locations on the playing field may also reveal individual strength and weaknesses during match play and allow investigating athletes' performances during training. This is particularly relevant for amateur players, as they often have considerable potential for improvement. Consequently extracting the player's locations over time from images of a single camera video feed is of particular interest, since they provide information for motion and game tactics analysis in competition and for training assessment.

In preliminary work, we have shown that human pose estimation using convolutional neural networks is able detect athletes' feet in squash in image coordinates and can be used for further processing [15]. To estimate positions from these coordinates on the planar playing field, geometric camera calibration is necessary.

For that, our application specific requirements for a camera calibration method are fivefold:

- R0: Allows for retrospective calibration of existing recordings
- R1: Accurate w.r.t. reprojection error on the game field and robust w.r.t. initial values
- R2: Fully automatic without manually defining point correspondences and no additional calibration target
- R3: Inexpensive hard and software
- R4: Mobile and fast setup

While several standard methods for camera calibration exist, they fall short fulfilling all our requirements. A distinctive feature of our specific application is the fact that the squash court itself can be regarded as a 3D calibration object. The obvious choice for camera calibration methods points at early methods using 3D calibration objects instead of planar calibration targets and manually establishing point correspondences. A naive solution for obtaining point correspondences, would require the user to click on junctions of the court lines in a certain order. However, ease-of-use (R2) is important for our application and we therefore aim for developing a fully automatic calibration method using a squash court as calibration object without manually provided point correspondences.

Geometric camera calibration can be posed as a multivariate non-linear iterative optimization problem: Finding a set of camera parameters that minimize an error function, which is usually the L2-distance between control points from the image and the corresponding reprojected control points from the calibration target. An accurate and robust calibration method provides good accuracy (low reprojection error), while allowing for poor initial values, respectively far away starting positions. Therefore, we investigate the field of genetic algorithms (GA) for optimization as they are known to find high-quality solutions even to NP-hard problems [16]. Thereby, we systematically evaluate which combination of fitness computation and genetic operator work best for accurate and robust camera calibration in our application context.

II. RELATED WORK

In early motion research studies, position notation was done fully manually [17]. This implies recording a full squash match followed by a tedious frame by frame video analysis. This approach is found not only in squash, but was also the usual approach in other racket sports e.g. badminton. A review is provided in [18] which compares manual, indoor / outdoor automated, and commercially available vision based analysis systems with respect to different kinds of sport. The authors conclude that with increasing computerization and computational power, the capturing process has become more and more (semi-) automated.

As an example for squash, the SAGIT / Squash software [19], a supervised automatic system for player tracking was introduced. This method requires a downward facing, ceiling mounted camera capturing the complete court from above and processes the captured images with a 384 px $\times$ 288 px resolution. For system calibration a custom radial distortion correction method as described in [20] was applied. Investigating the software's accuracy was done in [21]. This included, amongst other aspects, the calibration error. It has been reported that the players' systematic positioning error is higher for positions further away from the camera. Different

experiments showed that a player's position error is 0.08 m in the middle of the court and 0.33 m in the corners, defined as 0.5 m along each of two adjacent walls towards the middle. The authors state that these errors are related to imperfect calibration.

Different methods for camera calibration exist. One of the most common is Tsai's method [22] which is a two staged method based on a single set of coplanar image – world coordinates. First, the camera's 3D orientation, X – Y position, and scale factor are approximated. And subsequently in the second stage the focal length, distortion parameters, and Z position are estimated. An other well known calibration algorithm is Zhang's method [23]. In contrast to Tsai's method this requires a camera to observe a planar pattern from at least two (recommended are four to five) different orientations. Beside that, other camera calibration methods for live broadcasts are applied in the field of sports graphic systems. In [24] a method for calibrating multiple pan-tilt-zoom (PTZ) cameras is proposed. However this method is different to our approach as a PTZ camera's parameter changes constantly in contrast to our camera view. Further our method must be applicable to existing video recordings (R0), using single stationary camera views to allow for retrospective motion analysis. We also aim to be able to deploy our algorithm in courts with basic soft- and hardware requirements.

Beside that, first genetic algorithms for camera calibration already exist. In [25] an algorithm is proposed which relies on a pinhole camera model without modelling lens distortion. This method allows calibration using seven image – world coordinate correspondences. These calibration points are used for fitness evaluation of a possible solution. Compared to Tsai's method the authors state that their approach has a high accuracy and robustness even in noisy conditions. A second genetic approach including distortion correction is presented in [26]. Similar to [25] this approach also includes the process of extracting calibration points for estimating a solution's fitness.

Compared to the other, our approach is different in terms of fitness calculation. Instead of relying on the correspondences of the world image coordinates, we perform a lookup of a projection of the court in a distance-transformed edge image of the scene the fitness estimation. We also investigate different strategies for genetic operators in the algorithm.

We begin by describing the underlying pinhole camera model and show how geometry projection is performed. Subsequently we present our genetic algorithm, including our fitness estimation method and introduce all investigated operators. Then our experiments on synthetic and real world data are described and results are presented.

III. CAMERA MODEL

In general a camera model formally describes the image formation process of a physical camera, that is the mapping from a 3D scene to a 2D image. More specifically, it describes the projection of three dimensional (3D) world coordinates $[X, Y, Z]^T$ to two dimensional (2D) pixel coordinates $[u, v]^T$. For the basic pinhole camera model this projection is expressed formally in terms of (1) intrinsic and (2) extrinsic parameters [27]. The model's intrinsic parameters are defined by a 3×3 matrix:

$$\boldsymbol{A} = \begin{bmatrix} f_u & s & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

Matrix $\boldsymbol{A}$ contains a total of five unique camera parameters. The focal lengths f_u and f_v are given with respect to the corresponding image dimension in pixel units. Parameter s describes the skewness between the image's two main axes and models a shear distortion in the projected image. Since the shear distortion of modern cameras is close to zero, we neglect this parameter by setting $s = 0$. The remaining two parameters $[c_u, c_v]$ model the principal point offset. For perfect cameras this point corresponds to the image center. Together, the intrinsic parameter set is defined as $C_I = \{f_u, f_v, c_u, c_v\}$.

Next to the intrinsic parameters, the model's extrinsic parameters are defined by an augmented 3×4 matrix:

$$[\boldsymbol{R}|\boldsymbol{t}] = \left[\begin{array}{ccc|c} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{array}\right] \tag{2}$$

The first matrix $\boldsymbol{R}$ of the augmented matrix describes the rotation by a general 3×3 matrix. Although $\boldsymbol{R}$ contains in total nine entries, it is important to note that this is constructed by only three different, individual rotation matrices, each with a single degree of freedom. Thereby each of them specifies an elemental rotation around one of the camera's axes, i.e. α rotates around the x-, β around the y-, and γ around the z-axis as follows:

$$\boldsymbol{R}_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{bmatrix} \tag{3}$$

$$\boldsymbol{R}_y(\beta) = \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\beta) \end{bmatrix} \tag{4}$$

$$\boldsymbol{R}_z(\gamma) = \begin{bmatrix} \cos(\gamma) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

By multiplying these three individual matrices, the result is the final matrix $\boldsymbol{R} = \boldsymbol{R}_z(\gamma)\boldsymbol{R}_y(\beta)\boldsymbol{R}_x(\alpha)$ which describes the camera's full rotation. Beside rotation, the second part in the augmented matrix formulates the camera's translation. Translation is done for any dimension along the three main coordinate axes and the translation vector $\boldsymbol{t}$ consists of three elements $[t_x, t_y, t_z]^T$. Therefore, the final augmented matrix $[\boldsymbol{R}|\boldsymbol{t}]$ encodes the camera's extrinsic parameters with six degrees of freedom as $C_E = \{t_x, t_y, t_z, \alpha, \beta, \gamma\}$.

By multiplying the intrinsic matrix $\boldsymbol{A}$ with the augmented extrinsic matrix $[\boldsymbol{R}|\boldsymbol{t}]$, a world point in homogeneous coordinates $\boldsymbol{w} = [X, Y, Z, 1]^T$ is projected to an image coordinate

$\boldsymbol{m}' = [x, y, z]^T$, with

$$\boldsymbol{m}' = \underbrace{\begin{bmatrix} f_u & 0 & c_u \\ 0 & f_v & c_v \\ 0 & 0 & 1 \end{bmatrix}}_{\boldsymbol{A}} \underbrace{\begin{bmatrix} r_{11} & r_{12} & r_{13} & t_x \\ r_{21} & r_{22} & r_{23} & t_y \\ r_{31} & r_{32} & r_{33} & t_z \end{bmatrix}}_{[\boldsymbol{R}|\boldsymbol{t}]} \boldsymbol{w} \quad (6)$$

After perspective division $[x/z, y/z, 1]^T$, the corresponding 2D point is denoted by $\boldsymbol{m} = [u, v]^T$.

The pinhole camera model above describes a perfect lens which only models rectilinear projections. In other words, this model does not include image distortions. In real cameras however, there are lenses that deviate from the rectilinear projection and therefore lens distortion is important for camera calibration and must be addressed. Often distortion is radially symmetrical due to the symmetric construction of a lens. This causes straight lines to appear curved which mainly occurs in two variants. First, positive radial distortion (referred as ''barrel''), which makes the image look convex and known as ''fisheye'' effect and second, negative radial distortion (referred as ''pincushion'') which results in the opposite effect and lets images appear concave. Radial distortion for an image point $\boldsymbol{m}$ can be modelled using Browns model [28] by using three different coefficients k_1, k_2, k_3 with $r^2 = u^2 + v^2$ as follows:

$$\boldsymbol{m}_r = (1 + k_1 r^2 + k_2 r^4 + k_3 r^6)\boldsymbol{m} \quad (7)$$

An other type of distortion occurs when the lens is not perfectly parallel aligned to the image plane. This is called tangential distortion and results in tilted and stretched images. Tangential distortion is also included in [28] and modelled by using two parameters p_1, p_2 as follows:

$$\boldsymbol{m}_t = \begin{bmatrix} 2p_1 \cdot u\, v + p_2 \cdot (r^2 + 2\, u^2) \\ 2p_2 \cdot u\, v + p_1 \cdot (r^2 + 2\, v^2) \end{bmatrix} \quad (8)$$

The full distortion is then constructed using a set of five parameters $C_D = \{k_1, k_2, k_3, p_1, p_2\}$ and is composed as the sum of both distortion types $\boldsymbol{m}_d = \boldsymbol{m}_r + \boldsymbol{m}_t$. These distortion parameter set together with the previously described intrinsic and extrinsic parameter sets yield a 15-parameter camera model $C = C_I \cup C_E \cup C_D$ that allows the projection of an object in 3D space onto the 2D image plane of the camera.

IV. GEOMETRIC CAMERA CALIBRATION

Geometric camera calibration is a method for finding values for the parameters in C of a physical camera. The projection is modeled according to a camera model, such as the pinhole model (6) and may also include image distortions (7) – (8). The parameters of a physical camera describe how the camera captures a 3D scene onto a digital image, resp. sequence of digital images. In the same way these digital cameras capture objects of real 3D scenes, methods from computer graphics define synthetic cameras using the same pinhole model to render synthetic images of mathematically defined 3D objects. Early basic calibration methods combined images of a physical and synthetic camera, depicting the same physical 3D object of known geometry (e.g. a machined calibration target). Given a set of initial values, this allows for calibrating the camera by iterative refining the parameters:

Let $C_p \subseteq C$ and $C_s \subseteq C$ be two sets of initial values for a physical and synthetic camera, respectively. Let W be a set of 3D control points in world coordinates of a calibration target and M the corresponding set of image coordinates identified in an image I of the same calibration target obtained by camera C_p. Further, let $e(M_1, M_2) \in \mathbb{R}$ an error function like e.g. the euclidean distance metric on pairs of image coordinates. Then a calibration procedure is obtained by the following steps:

1) Project all $w \in W$ using (6) – (8) with C_s to obtain M'
2) Compute the projection error $e(M, M')$
3) Minimize e by adjusting C_s
4) Repeat with refined parameters C_s

In this process, step 3 is essentially an optimization method with respect to minimizing the re-projection error. For a perfect physical camera and convergence to the global minimum the error approaches zero and the camera's exact parameters C_p are given by the final adjusted synthetic camera C_s.

State-of-the-art methods rely on manually or automatically defined point correspondences between W and M as well as a precisely constructed calibration target. Our method is based on steps 1-4 but differs from related work in that we do not require a manually machined calibration target, nor do we require point to point correspondences. Instead, we regard an object that is anyway shown in the image as calibration target and establish world to image correspondences by means of a distance map of an edge image of the captured squash court. Furthermore, we propose a genetic algorithm for robust and accurate optimization.

V. CALIBRATION TARGET AND COURT GEOMETRY

The boundary of objects in 3D space can be represented, or at least approximated, by a set of connected spatial coordinates. In many sports, the playing field forms a boundary and can be defined by a network of corner points, connected by lines. In squash, the playing field (court) is a 3D object which is almost always visible in broadcasts and video recordings, except for close-ups and special camera angles. It is defined by five planes: four walls and one floor. The intersections of these planes define corner points and lines. Additionally, lines relevant to the game, are present on the planes, which also lead to more points. Due to the guaranteed visibility of the court in the captured image and its known 3D structure and dimensions, it can be considered a calibration target. The geometry of an object with known dimensions in world space can conveniently be represented by an undirected and unweighted graph:

$$G = (V, E) \quad (9)$$

$$V = \{[X, Y, Z]^T \mid X, Y, Z \in \mathbb{R}\} \quad (10)$$

$$E \subseteq \{(Q, S) \mid (Q, S) \in V^2 \wedge Q \neq S\} \quad (11)$$

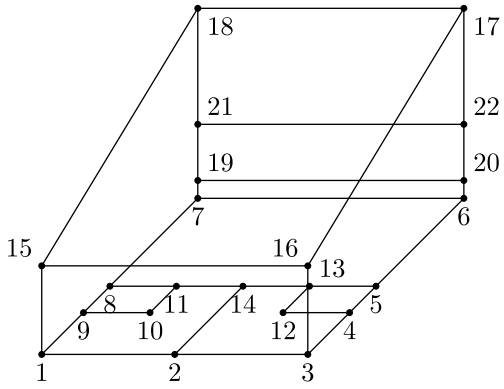

FIGURE 1. A standard squash court geometry can be represented as undirected, unweighted graph. In total it is constructed of 22 nodes and 32 edges.

Here, we define squash court geometry as a graph G with 22 nodes V and 32 edges E. Each individual node carries the individual location in 3D world space as information $[X, Y, Z]^T$. Edges represent connections between two nodes Q and S each, which form the court's boundaries and game defining lines. By rendering the court geometry using a synthetic camera, a visual representation of the projected court is obtained and used in our calibration method.

More specifically, we use the court geometry inside our fitness function to calculate the accuracy of the synthetic projection of the court geometry with the unknown projection of a given camera. The following sections describe a genetic algorithm (GA) with different operators in the context of optimizing camera parameters in order to minimize an error metric which is based on the distance transform.

VI. GENETIC ALGORITHM

Genetic algorithms (GA) as subclass of evolutionary algorithms (EA) are commonly used for solving optimization or search problems. As GA mimics evolutionary processes, the terminology used in this context is adopted from evolutionary biology. In general, all GA use the following building blocks: (A) For a given problem, a population of individual solutions (genomes), called a generation, is constructed. (B) A fitness metric, computed for each genome in the current generation. This metric states how well a genome is adapted to the problem to be solved. (C) Based on that, genomes are chosen by a selection process and used by (D) a crossover operator to produce two offspring. The main concept at this point is to ensure survival of the fittest, which is expected to result in a satisfying solution. However, to represent diversity through generations and improve the solution quality, there is the possibility for (E) genome mutation in the offspring. This selection and mutation process is repeated until the next generation contains enough genomes, so that the amount of individuals matches the initial population size. Now, the next generation represents the population in the next iteration which starts over again with fitness computation. This process is repeated until a termination criterion (F) is satisfied.

Our approach defines one population as a set of genomes representing cameras, obtained during initialization. The camera introduced in section III is given by 15 parameters and likewise a formal representation of one genome is given by a vector:

$$[f_u, c_u, c_v, c, t_x, t_y, t_z, \alpha, \beta, \gamma, k_1, k_2, p_1, p_2, k_3] \quad (12)$$

Please note the different order of distortion coefficients. We adopted this order to reflect the implementation which is based on OpenCV's application programming interface.

Algorithm 1

1:	Initialize N genomes as generation	VI-A
2:	Create the fitness map	VI-B
3:	**repeat**	
4:	Calculate fitness for genomes in generation	VI-B
5:	Sort genomes in generation desc. w.r.t. fitness	
6:	Add two best fitted genomes to next generation	
7:	**repeat**	
8:	Select two parents from generation	VI-C
9:	Crossover parents to produce offspring	VI-D
10:	Mutate option for both offspring	VI-E
11:	Add (mutated) offspring to next generation	
12:	**until** Next generation contains N genomes	
13:	Next generation becomes current generation	
14:	**until** Termination criterion satisfied	VI-F

Algorithm 1 outlines the proposed GA for camera calibration. In this context fitness is a measure for how well the projection of the synthetic camera matches the projection of the physical camera. Computing fitness does not require world to image correspondences of known calibration points. Instead, we look up values in a map, representing a distance transform of the image acquired by the physical camera. This distance map is constructed once at the beginning and is reused for fitness computation during the entire evolution process. An initialization function is used to create the first generation with N genomes. For computing the fitness of a single genome, the defined synthetic camera projects the geometry and the result is looked up in the distance map. Subsequently we sort the generation's genomes in descending order w.r.t. their fitness. Then the two best fitted genomes are retained for the next generation, which models a slight elitism selection. Genomes are then selected in pairs based on their fitness and two offspring are constructed by applying the crossover operator and mutated with a certain probability. This selection, crossover, mutation process is repeated until the next generation matches the current population's size. Finally the next generation becomes the current generation. In the following sections we investigate and adapt different strategies for the main GA-operators (A) - (F).

A. INITIALIZATION

During the initialization stage a number of genomes is defined and a population is constructed. A start genome

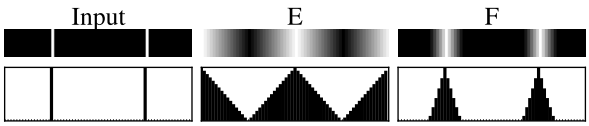

FIGURE 2. Top: Input image with two sharp edges, the associated distance transform E encodes distances to edges and the final lookup map F which shows the fitness. Bottom: Corresponding intensity profiles along a single pixel row.

as defined in (12) is used to derive an initial population with N genomes. For all parameters, besides the distortion parameters, offsets are added as random values drawn from a uniform distribution with individual ranges, resp. limits. Thereby, all genomes in the first generation share the same distortion parameters but differ w.r.t. all other parameters. The focal length offset limit is set to $\pm$ 100, the principal point offset range is $\pm$ 10. For all rotation parameters the limits are $\pm\pi/180$rad. While for X and Y translation ranges are $\pm$0.1 m, whereas for Z the range is $\pm$0.5 m. This distinction in translation parameters has been made to cover a wider range in the Z direction within the initial population. In this way, a population of N genomes around the initial value (starting point) in search space is constructed.

B. FITNESS

Determining for each genome an individual fitness score is an essential step in GA as this metric is the fundamental basis for controlling algorithmic evolution. First, to compute the fitness for an individual genome a fitness map $\boldsymbol{E}$ is extracted from the scene. To construct this map, initially an image captured by the physical camera (i.e. the unknown target genome) of the world scene is needed. As a first step this image is transformed to an 8-bit single channel edge image by using any known edge detector. Subsequently the distance transform, as described in [29] is applied by using a Moore neighborhood with radius 1 and Euclidean L_2-Norm with the original suggested values for $a = 0.955$ and $b = 1.3693$. As the resulting values in the map measure the distance in pixel, normalization to [0, 1] is performed. To increase the gradient's steepness of the normalized distance image $\boldsymbol{E}$, the lookup map $\boldsymbol{F}$ is computed by a scaled logarithmic transformation with values clamped to [0, 1] by:

$$\boldsymbol{F} = 1 - \max(0, \min(\ln(\boldsymbol{E} + 1)/0.3, 1)) \tag{13}$$

As shown in Fig. 2, transformation results in a border around a target edge. Thus, a pixel value in $\boldsymbol{E}$ reflects the distance at that location to the closest edge pixel in the target image. Finally, $\boldsymbol{F}$ allows for estimating the fitness, i.e. how well edges in the synthetic camera match edges in the target camera.

1) DISTANCE MAP

A first strategy for obtaining a fitness score is based on the projection of the 3D target geometry shown in Fig. 1 using a single genome of a generation. After thresholding, a binary image is obtained and the fitness value λ is computed as the sum of all rendered pixels in the lookup map. This is done by applying the binary image of the court as a mask to the distance map and summing up all non-zero pixel values. If the court in the binary mask is far away from the court in the distance map, the fitness value yields zero. On the other extreme, for a perfect match the fitness value is the maximum. In general, the better the match, the higher the fitness.

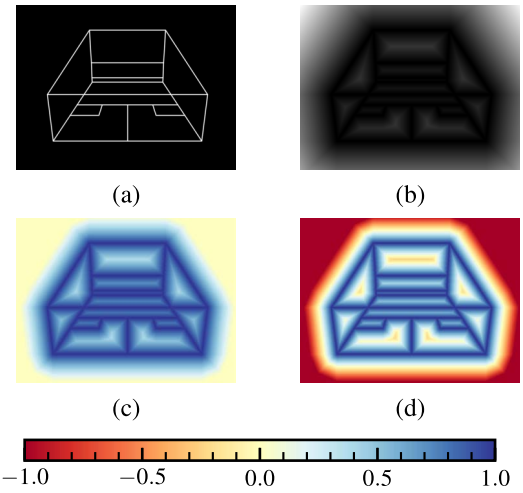

FIGURE 3. Comparison of two different fitness maps. (a) shows the binarized projection of a standard squash court using a camera. (b) shows the distance transform (E) of (a) using a Moore neighborhood with radius 1 together with the Euclidean L_2-Norm. (c) shows distance map F obtained from (b), containing only positive values. (d) shows distance map $2F - 1$, additionally containing negative values to model a punishment mechanism.

2) TRANSFORMED DISTANCE MAP

Our second strategy extends the first by punishing large deviations between the two courts by introducing an additional step during fitness map initialization. By computing $2\boldsymbol{F} - 1$, fitness lookup values are mapped from [0, 1] to [−1, 1] which results in a kind of punishment mechanism during lookup. Accordingly, a perfect projection still leads to a maximal fitness value, whereas a bad match / fitness yields a negative value in the worst case. Fig. 3 illustrates for the 3D court model shown in Fig. 1, a binary projection using a camera, the corresponding distance transform, as well as the first and second strategies of constructing the corresponding fitness maps.

C. SELECTION

In this phase of the algorithm reproduction of the current generation is initiated in order to create the next generation. To decide which individuals are used to produce the next population of desirably better cameras, the selection operator is applied. To ensure that always two fitted individuals are present in the next generation, we determine the two best fitted genomes and add them to the next generation. This models a very light elitism selection approach as we are retaining only

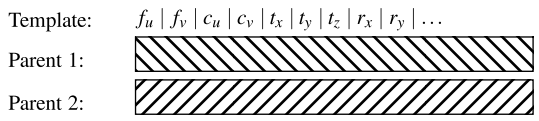

FIGURE 4. Simplified representation of two parent genomes (cameras). As indicated by the template, each parent consists of all camera parameters in C. A different hatching style represents a different camera.

the two best results, while the rest of the population is filled by repetitively applying one of the following operators:

1) ROULETTE WHEEL

The roulette wheel is the first strategy investigated. It performs a statistically weighted selection for two genomes of the current population. Here, the individual fitness values are used as the relative weights. Thus, a higher fitness results in a higher probability of being selected. Since this is performed on the same population twice per repetition, it is possible to select the same individual in both cases i.e. the two selected parents are the same.

2) TOURNAMENT

The selection of individuals based on a tournament mechanism depends on a tournament size k. First, a group is formed by uniformly selecting k individuals from the population as participants. While preserving their order w.r.t. descending fitness, the probability of being selected is $p(1-p)^a$ for the a-th participant. This ensures that participants with a higher fitness are more likely to be chosen. But compared to the roulette wheel it is possible that all participants show medium to low fitness values. In this way, even less adapted genomes have a chance to reproduce, favoring the best within competition. By using the tournament two participants are selected for crossover in the next step. We study tournaments with group sizes of $k = 4$ and $k = 8$ with $p = 0.5$ in both cases.

D. CROSSOVER

Using the result of any selection operator, the two selected genomes are interpreted as parents. As shown in Fig. 4 the parents are basically a genome (camera vector) with individual parameters. During crossover, they are recombined to produce two offspring which will replace them in the upcoming generation. In this way, both offspring carry the information from their parents over time. This is achieved by applying one of the crossover operators

1) SINGLE POINT

When recombining two parents using the single point strategy both genomes are split at the same random index, resulting in two sections per parent. The single point strategy denotes recombining the individual sections, by simply swapping the respective first sections. The resulting two children are shown in Fig. 5.

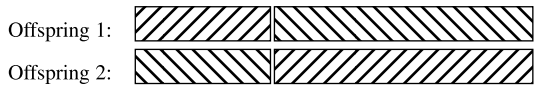

FIGURE 5. For single point crossover, the parents' parameters are split at a single location and are then individually used for both offspring.

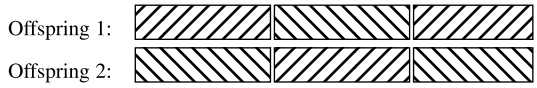

FIGURE 6. For two point crossover, the parents' parameters are split at a two locations and are then alternately used for both offspring.

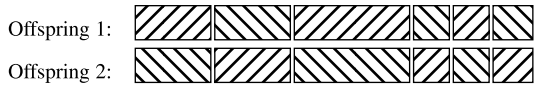

FIGURE 7. For uniform crossover, every parameter has its individual probability to be chosen from the first parent. This leads to a more divided distribution of the parents' parameters.

2) TWO POINT

Recombination involving two points works nearly identical to the single point strategy. Instead of splitting the genome into two parts at a random index, two random locations are used to obtain three sections per parent. By alternately combining these sections as shown in Fig. 6, two offspring are created.

3) UNIFORM

Compared to the previous strategies for crossover, uniform selection does not split the genome into several sections for recombination, although it may seem that way. Instead, for the first child genome values are selected independently from the first parent with a certain probability p. For the second child, the discarded values are used. Thus, this strategy corresponds to mixing the genomes together, where the probability can be interpreted as a factor influencing the mixing extent. When using this crossover strategy, we use a probability of either $p = 0.5$ or $p = 0.01$ for mixing genomes.

E. MUTATION

The mutation operator implements random modifications of genomes. This introduces diversity for genome values after crossover across generations and thus is intended to prevent the convergence to a local optimum. Although the presence of mutation is desirable and an important aspect, the mutation probability P_{mut} should be kept low, as high values lead to a rather random search. We apply mutation by offsetting every genome parameter with a statistically chosen value at a certain probability. In total we investigate two different types of mutation:

1) UNIFORM MUTATION

The first mutation strategy utilizes continuous uniform distributions $U(a, b)$. Every genome parameter is defined separately, thus the limits are defined individually. If a genome

TABLE 1. Applied distributions and parameters for mutation operators according to strategies. Translation and rotation parameters differ by their distribution in the distribution mutation strategy.

Parameter	P_{mut}	Uniform mutation	Distribution mutation	Offset
f_u, f_v	0.05	$U(-10, 10)$	$U(-10, 10)$	0
c_u, c_v	0.05	$U(-50, 50)$	$U(-50, 50)$	0
t_x	0.05	$U(-0.1, 0.1)$	$N(0, 0.1)$	0
t_y	0.05	$U(-0.15, 0.15)$	$N(0, 0.2)$	0
t_z	0.05	$U(-0.5, 0.5)$	$LN(0, 0.7)$	-1
α	0.05	$U(-0.05, 0.05)$	$N(0, 0.05)$	0
β, γ	0.05	$U(-0.05, 0.05)$	$N(0, 0.01)$	0
k_1, k_2	0.01	$U(-0.5, 0.5)$	$U(-0.5, 0.5)$	0
p_1	0.01	$U(-0.2, 0.2)$	$U(-0.2, 0.2)$	0
p_2	0.01	$U(-0.1, 0.1)$	$U(-0.1, 0.1)$	0
k_3	0.01	$U(-3, 3)$	$U(-3, 3)$	0

TABLE 2. Operators with associated available strategies used for composing the strategy bundles. Values for selected strategy parameters are shown next to them.

Operator	Strategy	Strat. Parameters
Initialization	Uniform	$N = 8, N = 16$
Fitness	Distance map (DM)	N/A
	Transformed DM	N/A
Selection	Roulette wheel	N/A
	Tournament	$k \in \{4, 8\}, p = 0.5$
Crossover	Single point	N/A
	Two point	N/A
	Uniform	$p \in \{0.5, 0.01\}$
Mutation	Uniform	N/A
	Distribution based	see Table 1
Termination	No improvement	$n = 300$

parameter is selected for mutation, a value is drawn from the corresponding distribution and subsequently added to the corresponding genomes value. While we set the mutation probability for a parameter to be chosen for all intrinsic and extrinsic values to $P_{\text{mut}} = 0.05$, the probability for distortion parameters is $P_{\text{mut}} = 0.01$. We implement this strategy using the distributions shown in Table 1.

2) DISTRIBUTION MUTATION

This strategy uses different types of distributions (uniform, normal, lognormal) for mutating genome parameters, instead of simply using the uniform distribution of different ranges. If a parameter was chosen for mutation based on the specific probability P_{mut}, a value is drawn from the selected distribution instead and subsequently added to the parameter's value. In addition we implemented shifting the chosen value in a fixed direction. For that a defined scalar offset is added. While this would obviously be redundant for a normal distribution $N(\mu, \sigma^2)$ as one could achieve the same effect by changing μ, it is however a convenient possibility for shifting values drawn from a lognormal distribution. While retaining uniform distributions for intrinsic and distortion, we changed the distributions for all extrinsic parameters for this strategy. Values for all rotation and t_x, t_y translation parameters are selected using normal distributions. The only parameter that is assigned a lognormal distribution is t_z as the probability density function (PDF) of this distribution appears skewed. Accordingly, values in a specific direction can be preferred. As for squash recordings it is common that the court fills the scene, translation w.r.t. t_z may be also preferable in a specific direction. All used distribution types are shown in Table 1 with their respective parameters.

F. TERMINATION

Checking the binary termination condition is the final step in the GA during every iteration. The result determines whether the current generation of the algorithm was the last or whether to proceeded with another generation. Different strategies can be used according to the application at hand for various reasons. If the application is time critical, a fixed time bound can be specified and the best fitted genome up to that elapsed time is assumed to be the solution. This leads to the problem that on different machines, the number of generations for finding solutions vary. And due to this, it can lead to bias and lack of comparability of results. Another option for addressing this problem is to use a fixed number of generations. However, finding the best amount of generations is difficult, because a too high value consumes unnecessary computing time and a too low value does not find an optimum. Following that a more convenient strategy would be to define a sufficient fitness value. Unfortunately, by using our fitness strategies the best fitness value is unknown and therefore no good assumption for an acceptable fitness value can be made. Therefore, in our method as soon as the best fitness value stagnates for a given number of generations, the corresponding genome is assumed to be the best fitted solution for the problem. In our case we terminate when the fitness stagnates for n $=$ 300 generations, i.e. the last improvement was observed 300 generations ago. For the sake of completeness, it should be mentioned that various logical conjunctions from different strategies are also possible but not investigated further.

VII. EXPERIMENTS

As laid out in sections VI-B – VI-F, various operators (e.g. fitness, selection, mutation) and for each operator various strategies are available to drive the evolutionary process, simulated by our genetic algorithm. We define the combination composed of one selected strategy for each of the six operators as a strategy bundle. Table 2 lists for each operator the strategies considered. With 2 different initialization-, 2 fitness-, 3 selection-, 4 crossover-, 2 mutation-, and 1 termination strategy variations, a total of 96 unique strategy bundles are available. For evaluation purposes, we have conducted experiments that systematically execute all possible strategy bundles on synthetically created data.

To compute and compare performances, for each experiment a target vector, representing the physical camera is defined which the GA should approach during optimization. Values for this vector are based on a camera typically found in squash recordings.

TABLE 3. Offsets for deriving a start camera from a given target camera. Three offsets per parameter group result in near, medium, and far distances, respectively low, medium, and high optimization complexity.

Parameter		Near	Medium	Far
f_u, f_v, c_u, c_v	(px)	± 10	± 50	± 100
t_x, t_z	(m)	± 0.1	± 0.5	± 1.0
t_y	(m)	± 0.1	± 0.5	± 2.0
α, β, γ	(rad)	± 0.01	± 0.1	± 0.2
k_1, k_2		± 0.1	± 0.25	± 0.5
p_1		± 0.025	± 0.1	± 0.2
p_2		± 0.01	± 0.05	± 0.1
k_3		± 0.5	± 1.5	± 3.0

In addition to the target vector, another vector representing a camera vector to start the optimization with, is required for each experiment. Since optimization complexity highly depends on the euclidean distance between target and start vector, we control complexity by simply adjusting the start vector to obtain a target vector. For that, we define and apply a wiggle operator to $\lfloor 15/2 \rfloor = 7$ randomly chosen parameters of the camera model. The operator is able to alter a camera vector in three different modes by utilizing one of the offsets per parameter shown in Table 3. For each chosen parameter either a positive or negative offset is randomly selected and added to the corresponding target's parameter value. In that way we vary $\approx 50\%$ of the target's parameters to obtain the start vector, resulting in low, medium, or high optimization complexity.

Thereby, the wiggle operator allows for randomized, but controlled construction of the start genome. Applying each operator mode to each of the 96 strategy bundles leads to a total of 288 unique experiments.

Since an edge image of the court geometry serves as input to the fitness map, we consider different quality levels for the computed edge image. To that extent, we add 5 different types of noise to the court projection before computing the fitness: First we add salt as a local impulsive noise. Second, horizontal and third vertical lines are added to model false positively extracted edges. The fourth variation combines horizontal and vertical lines, forming a regular grid. Lastly, rotating the grid by 45° represents the fifth variant. Fig. 8 shows fitness maps for the same court projection, but with previously added noise types.

In summary, these noise variations allow for investigating how inaccuracies during the initial edge extraction affect fitness computation. Additionally considering six noise types (including no noise), the number of unique experiments extends to 1728.

To ensure that results are not random, we repeat each of the 1728 unique experiment 32 times, resulting in a total of 55296 optimizations were performed. While no change to the experiment's strategy bundle and noise type was made, a new starting vector was constructed for each optimization.

During optimization, we keep track of several performance metrics. Beside the number of generations until convergence, we compute the reprojection error for three different geometries. As we defined in (9) – (11), a single geometric object can be represented as a graph G. After projecting all vertices of a graph to image coordinates, the projection error is computed as the euclidean distance of the true image point m, obtained by the target camera and $\hat{m}$, obtained by the result camera. The first geometry G_c we consider is the court itself. From this a geometry G_b is derived by sampling the bounding box in all three dimensions. Finally, we define a third geometry $G_p^{m \times n}$ as a regular $m \times n$ sampled grid, representing vertices on the court's playing field. This one is of particular interest, since athletes feet are usually located on the floor, therefore the accuracy on this surface is crucial for movement analysis.

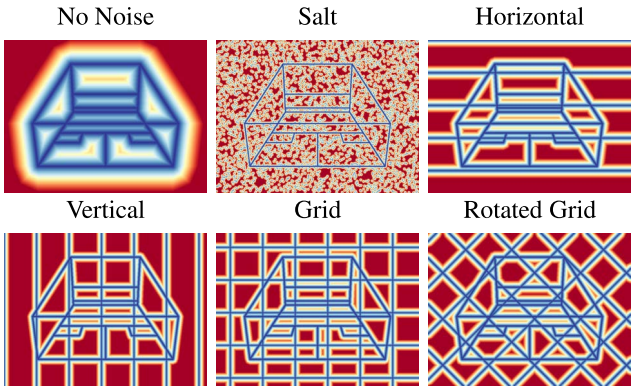

FIGURE 8. Fitness maps for an edge image with different artificial noise to model possible inaccurate edge extraction.

A. EXPERIMENTAL RESULTS

1) MEAN REPROJECTION ERROR FOR DISTANCE / NOISE TYPES

First, we compare the three start distances and the six noise types to one another. For that we use the mean reprojection error (MRE) of the court's playing field geometry $G_p^{16 \times 16}$, which includes in total 256 vertices on the court's floor. Sorting all MRE results in ascending order and computing the normalized cumulative sum w.r.t. distance (18432 each) and noise type (9216 each) investigation of the number of optimizations with MRE below a threshold is possible. Fig. 9, shows results for the distance type and the noise type separately, up to a threshold of 10 px.

By comparing the distance type results, it can be seen that 60% of near optimizations show an MRE of < 2 px. At 8 px, the proportion increases to 91%, while at this threshold medium distance experiments reach 38%. The far distance experiments first reach 40% at a very high MRE of 50 px, while at this threshold already 99% of the near and 82% of the medium experiments can be found.

Regarding noise variation, experiments with horizontal and vertical line noises reach 23% and 25% respectively for an MRE < 2 px. Up to this threshold, 23% of both grid variants are located while 17% are observed for salt. With no noise only 19% show an MRE < 2 px. Considering an MRE threshold of 4 px, 47% of the noiseless, 36% and 40% of the horizontal and vertical, and 36% of both grid types can

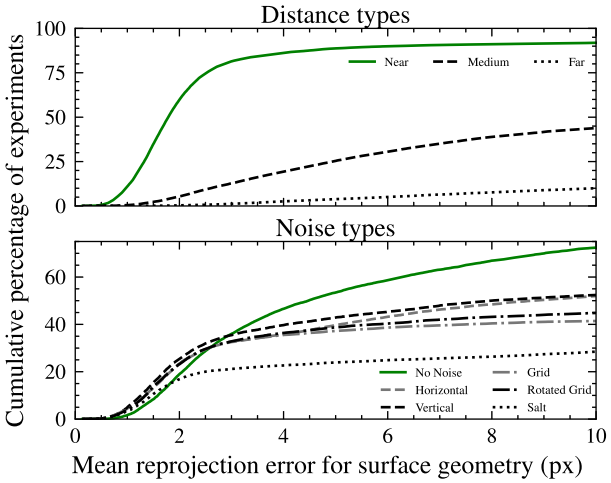

FIGURE 9. Relative number of experiments below a given mean reprojection error (MRE) threshold as a function of MRE threshold. Results with respect to distance types and all six noise types are shown at the top and bottom, respectively. Note the different ranges of the vertical axes.

be found. Here, salt shows the lowest experiment proportion with 23%. Even considering an MRE of 6 px shows only a 24% for salt experiments, whereas noiseless achieves 59%, horizontal and vertical 43% and 45%, respectively.

In general, two conclusions can be drawn from the results. First, unsurprisingly, the distance between start and the target vector is a key factor, since with smaller variation in camera parameters a higher fraction of experiments already shows a low MRE. Secondly, it can be concluded that optimization quality is directly related to the quality of the edge image. Although noise in terms of horizontal, vertical, or grid lines decrease the optimization quality compared to perfect noiseless edge images, the worst results are shown for salt noise experiments. These findings support that large proportions of high frequent noise influence the result the most. Therefore, it can be concluded that these effect should be avoided during edge detection or should be filtered beforehand.

2) NUMBER OF GENERATIONS

To compare computation time, we consider the number of evaluated generations until termination. Although our termination strategy is based on the fact that fitness stagnates for 300 generations, and thus the result is already found earlier, these generations are part of the applied GA and are therefore purposely included in this metric. We group experimental results w.r.t. distance and noise types and report their distribution in Table 4. Each group theoretically consists of n = 3072 experiments, but as we only consider experiments with a $G_p^{16\times16}$ MRE of < 10 px (26860 in total) the number of experiments is different per group.

For every noise type individually, it can be seen that with increasing complexity, the number of generations also increases. Here, salt again stands out in particular, since the rate of generation increase is lower compared to other

TABLE 4. Median number of generations until termination as a function of distance and noise type for experiments with a mean reprojection error of < 10 px on the court's playing field.

	Near		Medium		Far	
	Generations (n)		Generations (n)		Generations (n)	
No noise	894	(3071)	1145	(2547)	1296	(1055)
Horizontal lines	976	(3032)	1202	(1472)	1349	(285)
Vertical lines	1000	(3003)	1258	(1586)	1453	(240)
Grid	975	(2849)	1217	(890)	1386	(78)
45° Grid	984	(2797)	1233	(1178)	1468	(155)
Salt	1019	(2182)	1072	(411)	1068	(29)

noise types. This lets us conclude that salt noise has a large impact on the fitness computation, regardless the distance type. Further salt seems to distract the GA, leading to more generations.

To examine the computational cost, we measured the GA's mean run time per generation. All measurements were performed sequential in a single thread using non-optimized Python code on a standard CPU. We observe that with larger populations as well as with higher image resolution, the required computational effort increases. For an 800 px × 600 px fitness map, population sizes of 8, 16, and 32 achieve run times of 5.2 ms, 9.8 ms, and 21.8 ms per generation. For 1920 px × 1080 px, the run times increases to 18.4 ms, 32.9 ms, and 66.6 ms. This demonstrates a median run time of 32.9 s for 1000 GA generations with 16 individuals at a 1920 px × 1080 px (Full HD) resolution.

3) STRATEGY BUNDLES

For evaluation of the different strategy bundles, we consider all results on the $G_p^{16\times16}$ geometry with an MRE < 10 px. An inspection of strategy distributions shows that the best 28 strategy bundles include the transformed distance map for fitness computation. Among these, 21 are found with a population of 16 individuals. In 12 cases the roulette wheel is used for selection and 19 times the distribution mutation. For the crossover operator, no strategy stands out in particular.

From this we conclude that for the implemented GA the type of crossover has no particular influence, much more important is a larger population. Moreover, for the best results, the transformed distance map is crucial. In summary, we present the following combination to represent the best strategy:

(A) Initialization: 16 individuals
(B) Fitness: Transformed distance map
(C) Selection: Roulette wheel
(D) Crossover: e.g. SinglePoint
(E) Mutation: Distribution based mutation

4) FITNESS ERROR AS A FUNCTION OF TIME

Insights into the optimization process can be obtained from plotting the fitness error as a function of time, respectively number of generations. As for this synthetic experiment, the start and target camera vectors are known, the fitness error can be computed at any step of the optimization process. For

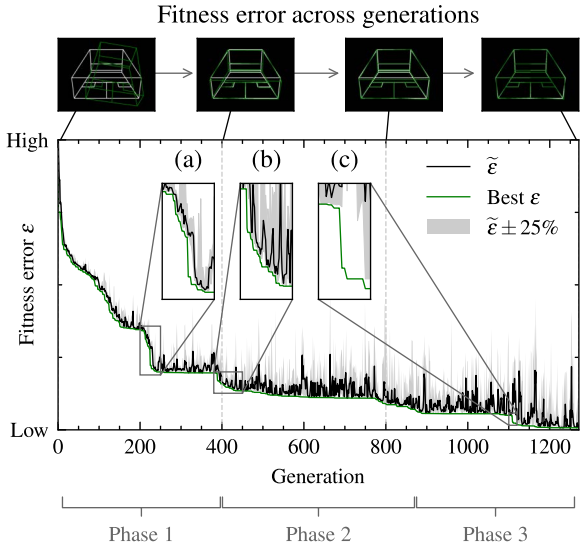

FIGURE 10. Fitness error as a function of time, illustrating the optimization process. Continuous variation and adaptation of the population increases fitness and thus decreases the fitness error. Next to incremental improvements and plateaus, large optimization steps can be observed in phases 1-3 ((a)-(c)), indicating that the GA algorithm repeatedly found its way out of local minima. Selected projections of the court geometry are shown in the top row. Please note that the final 300 generations are not shown, since the termination criterion was met, resulting in no improvement during this period.

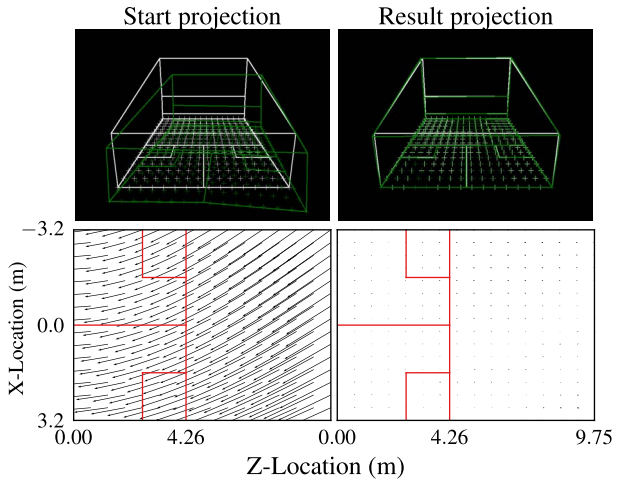

FIGURE 11. Projection error estimated from synthetic 3D grid points placed on the squash court's floor. Top: Court projections obtained from the start camera (left) and from the result camera (right). The unknown target geometry is shown in white, start and result in green. Bottom: Corresponding top views of the courts above, showing displacement vectors (arrows) of all grid points.

this, we run an optimization by using our best found strategy (VII-A3 A-E). Error measures λ and λ_{target} represents the fitness value obtained by projecting court geometry $G_p^{16\times16}$ using the camera vector corresponding to the current population and the target camera vector, respectively. Then, the fitness error is given by $\varepsilon = \lambda_{\text{target}} - \lambda$. Thus, the closer the current projection to the target projection, the lower the error. Fig. 10 illustrates the process of the population adapting to the target during the first 1300 generations. The top row shows two court geometries at four points in time, rendered by the current and target camera, respectively:

The fluctuations of the mean fitness error $\tilde{\varepsilon}$ suggests that the algorithm varies camera parameters and thereby explores the search space. This agile method eventually leads to convergence, respectively adaptation to the target geometry. In particular, the GA algorithm's ability to repeatedly escape local minima and thereby improving the solution becomes evident.

5) PROJECTION ERROR ON PLAYING FIELD

We aim for analyzing on-court player location and motion and therefore, the error of estimating 3D world locations from 2D image coordinates needs to be considered. For determining this error, we construct our camera model from a genome vector. We then compute a ray starting from the camera's focal point through the image point and compute the intersection with the court's floor. The top row of Fig. 11 shows exemplary projections of the court G_c and the playing field $G_p^{16\times16}$ obtained by the start (left) and result (right) genome. Below, the projection error for all 256 $G_p^{16\times16}$ sampling points is shown by displacement arrows. The results were obtained using our best identified strategy (VII-A3 A-E), without noise for a medium optimization distance.

In white the target vector was used to project the court G_c and playing field $G_p^{16\times16}$ geometry. While the left side additionally shows the projection obtained by the start vector, the right shows the final result projection in green, respectively. Below, the court is shown from above. Arrows indicate projection errors for all grid points, resulting from 3D estimation using the corresponding camera. After iterating for 4261 generations, a clearly visible overall improvement can be seen between the start and result genome. The resulting accuracy (short arrows) would provide sufficient accuracy for our application of position estimation in squash.

6) GA COMPARED TO CONVENTIONAL OPTIMIZATION METHODS

Since a GA performs optimization with respect to some objective function, it can be compared to standard optimization methods. Here, we use the Subplex [30] and DIRECT-L [31] methods, provided by the NLopt library [32]. The maximum number of fitness function evaluations per optimization was set to 100000. Following our GA experimental setup, optimizations were performed for every fitness map, distance, and noise types at a 32-fold repetition each. This results in a total of 2304 optimizations performed. Fig. 12 presents a comparison of all three optimization methods, which achieve a $C_p^{16\times16}$ MRE < 10 px.

In Fig. 12 a) it can be seen that for near distances Subplex with a median MRE (px) of 2.8 shows better results than DIRECT-L with 5.8. For medium distances, both algorithms are almost equal with 5.9 and 5.3, respectively. Last, for the

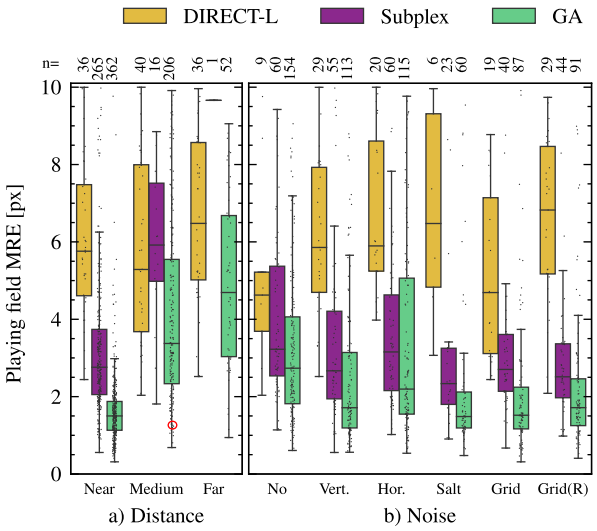

FIGURE 12. Distribution of MRE for DIRECT-L, Subplex, and GA with playing field MRE < 10 px. In a) results are shown with respect to distance and in b) for noise separately. The corresponding number of experiments *n* is given above each plot. The red circle indicates the result for the experiment presented in Fig. 11.

far distance, only a single Subplex optimization shows an MRE of < 10 px (9.7), while the median MRE for DIRECT-L is presented as 6.5. These results are explained by the fact that Subplex is intended for local and DIRECT-L on the other hand for global optimization. The GA reports, for all distance types, the lowest median MRE with 1.5, 3.4, and 4.7. When discriminating the experimental results w.r.t. noise, the median MRE shows a similar result. For all noise types presented in Fig. 12 b) DIRECT-L has the highest median MRE followed by Subplex optimization results. While again, the GA demonstrates the smallest median errors. It is further noticeable that a lower median MRE is measured for the GA experiments with salt noise compared to noiseless experiments. This seems to be contradictory to our previously reported results. However, here only the best strategy bundle is used whereas previous observations included all strategy combinations. This also can be seen by the number of experiments, which are reported with 60 for salt and 154 for no noise. Thus, the group of experiments, at an MRE < 10 px, without noise contains a significantly larger proportion compared to experiments with salt noise. We conclude that Subplex is capable of optimizing at short distances using our fitness lookup map, but becomes significantly worse as the distance increases. For all nine experiments GA outperforms the conventional methods.

B. REAL WORLD RESULTS

So far, our investigations and reported results are based on perfect images of the squash court geometry. They can be understood as the actually required edge images. Even though real world aspects have been modeled by adding different artificial noise types, actual real world scenarios include more complexity. Therefore, we apply the described GA in two other scenarios: (1) artificial renderings and actual (2) real-world images. In both cases edge extractions as a preprocessing step is required. In Fig. 13 our experimental results for four artificial renderings, six real-world situations are presented. Each result includes projections of the court's geometry G_c for its initial and final camera, overlaid on top of the physical camera image, as well as the extracted fitness map $\boldsymbol{F}$.

For (1) artificial scenarios, a single court with red lines was modeled using the 3D computer graphics software Blender [33]. If was then rendered in four different perspectives. Two of them are from a straight perspective, one depicting the entire court, while the other shows the court cropped across its top and bottom. The other two capture the entire court but with a Z-axis angle of $\pm 5°$, so that the court appears slightly rotated. These four perspectives were chosen as they represent the most common camera setups in squash broadcasting. Preprocessing was carried out by using Canny edge extraction [34] applied to the gray scale transformed image for all four images. The geometry used for optimization was limited to the visible edges only. Thus, computation of the fitness value is only based on actual extracted observations.

By using (2) real images we go one step further and apply the GA to images extracted from video recordings of real squash matches. As this is a retrospective investigation, we manually removed any existing athletes in the images by hand. This is done by subtracting different frames across the video, with athletes located at different positions. By that we obtain empty court images. Edge extraction was done in the same way as for the artificial renderings. This also includes limiting the geometry to the visible edges. We derive the start vector from the fact, that for broadcasts and video recordings of squash matches the camera position corresponds to a typical viewer position centered behind the court and therefore can be good estimated. This is done by averaging the obtained parameters of previously carried out camera calibration results.

Results for experiments of both scenarios (1) and (2) are shown in Fig. 13. In total three components make up each sub-image (a)-(j). The upper left depicts the start vector for projection which is overlaid on top of the physical camera image. Below that, the fitness map $\boldsymbol{F}$ computed after applying Canny edge detection is shown. Finally, on the right the projection of the GA's optimized camera is presented as overlay.

The (1) artificial images are shown in (a)-(d). First, it can be seen that fitness maps very clearly correspond to parts of the court geometry. We explain this by the fact that there is no disrupting background, no advertising, mirroring or similar effects present on these courts. Applying the GA shows good results for the fully captured courts (a)-(c). For the truncated court in (d), however, two problems exist. First, due to the front wall truncation, the geometry projection appears stretched horizontally in (I). The same effect can be seen at the bottom of the image, at the back wall. Since the lower

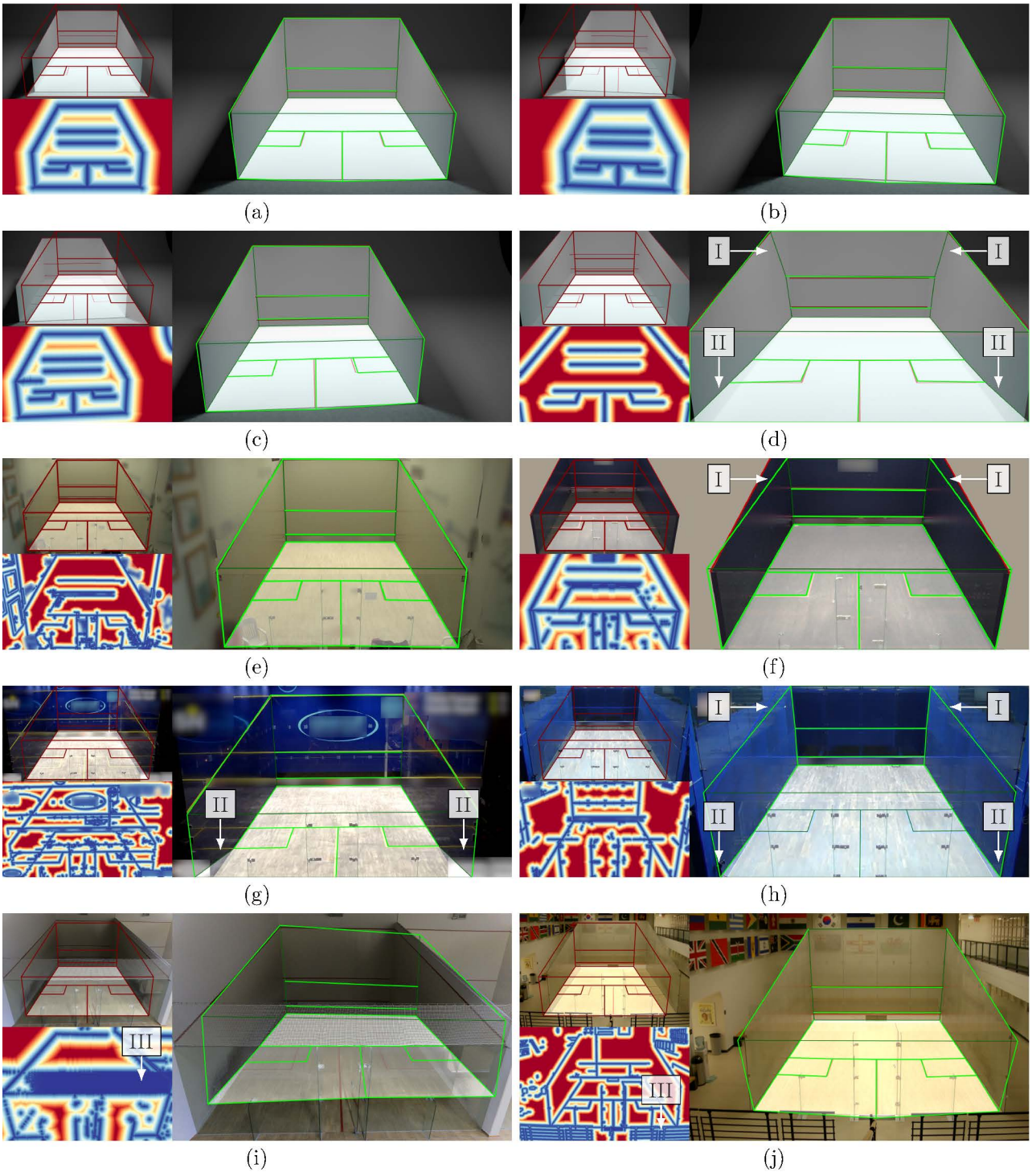

FIGURE 13. Fitness maps and geometry projections for the start and the GA optimized camera on artificial and real squash court scenarios, respectively. In (a)-(d) artificial and in (e)-(j) real world scenarios are presented. Each scenario is composed of three images. First, in the top left, geometry projection by the start camera (initial vector) is shown on top of the physical camera's captured image. Below, the fitness map obtained after edge detection is presented. Finally on the right, geometry projection by the optimized camera (result vector) is overlaid.

edge is not visible here, the available edges on the side in II get stretched and thus do not correspond to those captured by the physical camera.

The real world images (e)-(h) show a similar picture. Again, it is noticeable that a good result is achieved for a fully captured court (e). With a truncated front wall (f), again the misalignment (I) is present at the top of the image. But as no stretching occurs in the floor area, the projection for this area corresponds to the physical camera and is therefore sufficient. On the other hand, a cut off backwall (g),

the result gives a different picture. Similar to the artificial images, horizontal stretch (II) is present. Thus, the projection of the back court area does align with reality. Similar to the modelled situation (d), the court in (h) is cut off at the front as well as at the back wall. The result is a projection which is not fully accurate either in (I) nor in (II). All courts (e)-(h) considered so far, present realistic fitness maps as they appear after edge extraction. In contrast to the artificial ones, here edges extracted from advertisements and the like are also present. These results show that this has no influence on the adaptation process by the GA.

The situation for a weak fitness maps is different. An example is given in (i). A mesh above the backwall for keeping balls from flying off court, leads to many edges which are interpreted as high fitness values in the fitness map (III). Consequently, the camera projection adapts to a state where edges of the geometry are projected to exactly that region. The final result is then a fully misaligned camera. A similar effect is shown in (j), where a railing located at the bottom of the image leads to many edges. Although these represent high fitness values there is no misalignment regarding the final projection. This may be due to the fact that the interference is not directly within, but more outside of the geometry to be fitted. Additionally, here the court is not horizontally centered and instead slightly shifted to the right. The final projection shows that the GA is still able to manage this situation. From (i) and (j) can be concluded, that the GA is sensitive to large interference inside, but not outside the target geometry.

VIII. CONCLUSION

We presented a GA for camera resectioning in squash based on a known three dimensional court geometry. Further, we also implemented multiple variations of operators for the GA and evaluated the resulting possible strategies. On this basis, we have shown that the mean reprojection error on the court's playing surface is within an acceptable range when considering synthetic images with non-extreme starting conditions (R1). By translating these findings to artificial and even real world scenes, we have also shown that the GA is able to perform camera resectioning in those cases. However, this is only possible if edge extraction has been performed successfully beforehand. Following that, our results demonstrated to be fully automatic after edge extraction as no additional calibration target is needed (R2). As our implementation relies on consumer hard- and free software, we do not have any special requirements (R3). Our results show that run time performance is acceptable, but may be improved by a multiprocessing approach. Since only a fixed physical camera, capturing the empty court, is needed, mobile and fast camera setup is possible (R4). As we have demonstrated retrospective camera resectioning (Fig. 13) and conclude that our GA allows for retrospective analysis of existing recordings of squash matches (R0). Overall, we conclude that our specific requirements R0 - R4 are fulfilled.

Our analysis leads to the following additional conclusions: Since the court serves as a geometric calibration object, truncated courts affect the scenarios in which our approach is applicable. This is especially true in applications where accurate court floor projection is important (e.g. estimation athlete positions on the playing surface). False projections on the front wall are of minor interest in these scenarios. Contrary to this, incorrect geometrical fitting of the court's back wall is more problematic. In these cases no accurate positioning near the corners is possible. The main conclusion that can be drawn from this is that lastly the type of application matters. For example in scenarios where athlete position should only be assigned to large quadrants on the playing surface, our method is still very well suited. Whereas with decreasing quadrant sizes the applicability of our approach also decreases. This is based on the fact, that at certain locations, no or insufficiently accurate positional information can be obtained. Beside that, we have shown that for real world scenarios the quality of the extracted geometry, i.e. edge image, matters. This conclusion follows the fact that this extraction builds the foundation for the fitness map and fitness computation. On this basis, we conclude that the GA provides a well suited method for easy stationary camera resectioning, from which sports applications with an easy extractable and well defined geometry could benefit from.

A. FUTURE WORK

As edge detection is challenging, it is left for future research to systematically investigate different preprocessing steps for geometry extraction and ways to replace the basic canny approach. This future research should consider the potential effects for geometry extraction more carefully. Regardless, future research could continue to explore the implementation of a crossover operator based on logical groups: For example crossover of two offspring may be applied to a single parameter group (e.g. translation or rotation) only. Future research should certainly further test whether the GA can benefit from the implementation of evolutionary phases. As an example of this, it is probably worth investigating whether to limit fitting to parameter groups (e.g. translation or rotation) for a given number of generations. Rotating the parameter groups could be performed in round robin until the termination criterion is satisfied. So far, we have investigated false positive errors in edge extraction. In further studies, false negative errors should also be investigated. This can be achieved by reducing the edges of a geometry for fitness map generation while considering the full geometry for fitness calculation. It is a question of future research to compare our single camera approach to other sports multi camera calibration methods. In order to translate our findings into the real world, our next step is to apply the presented method for automatic camera resectioning to a squash specific training scenario for extracting the players' motions paths while exercising.

REFERENCES

[1] M. Bosch-Jorge, A.-J. Sánchez-Salmerón, Á. Valera, and C. Ricolfe-Viala, "Fall detection based on the gravity vector using a wide-angle camera," *Expert Syst. Appl.*, vol. 41, no. 17, pp. 7980–7986, Dec. 2014.

[2] M. Dimitrievski, P. Veelaert, and W. Philips, "Behavioral pedestrian tracking using a camera and LiDAR sensors on a moving vehicle," *Sensors*, vol. 19, no. 2, p. 391, 2019.
[3] F. Albiol, A. Corbi, and A. Albiol, "Geometrical calibration of X-ray imaging with RGB cameras for 3D reconstruction," *IEEE Trans. Med. Imag.*, vol. 35, no. 8, pp. 1952–1961, Aug. 2016.
[4] F. Albiol, A. Corbi, and A. Albiol, "Densitometric radiographic imaging with contour sensors," *IEEE Access*, vol. 7, pp. 18902–18914, 2019.
[5] C. Balletti, F. Guerra, V. Tsioukas, and P. Vernier, "Calibration of action cameras for photogrammetric purposes," *Sensors*, vol. 14, no. 9, pp. 17471–17490, Sep. 2014.
[6] T. N. Schoepflin and D. J. Dailey, "Dynamic camera calibration of roadside traffic management cameras for vehicle speed estimation," *IEEE Trans. Intell. Transp. Syst.*, vol. 4, no. 2, pp. 90–98, Jun. 2003.
[7] A. Corbi, O. C. Santos, and D. Burgos, "Intelligent framework for learning physics with aikido (Martial Art) and registered sensors," *Sensors*, vol. 19, no. 17, p. 3681, Aug. 2019.
[8] T. J. Gabbett, "The training—Injury prevention paradox: Should athletes be training smarter and harder?" *Brit. J. Sports Med.*, vol. 50, no. 5, pp. 273–280, 2016. [Online]. Available: https://bjsm.bmj.com/content/50/5/273
[9] T. W. Jones, B. K. Williams, C. Kilgallen, C. Horobeanu, B. C. Shillabeer, A. Murray, and M. Cardinale, "A review of the performance requirements of squash," *Int. J. Sports Sci. Coaching*, vol. 13, no. 6, pp. 1223–1232, Aug. 2018.
[10] World Squash Federation. (2016). *Specifications For Squash Courts*. [Online]. Available: https://www.worldsquash.org/wp-content/uploads/2017/11/171128_Court-Specifications.pdf
[11] M. Hughes and I. M. Franks, "Dynamic patterns of movement of squash players of different standards in winning and losing rallies," *Ergonomics*, vol. 37, no. 1, pp. 23–29, Jan. 1994.
[12] G. Vuckovic, B. Dezman, J. Pers, and S. Kovacic, "Motion analysis of the international and national rank squash players," in *Proc. 4th Int. Symp. Image Signal Process. Anal.*, Oct. 2005.
[13] G. Vučković, J. Perš, N. James, and M. Hughes, "Tactical use of the t area in squash by players of differing standard," *J. Sports Sci.*, vol. 27, no. 8, pp. 863–871, Jun. 2009.
[14] S. Murray, N. James, J. Perš, R. Mandeljc, and G. Vučković, "Using a situation awareness approach to determine decision-making behaviour in squash," *J. Sports Sci.*, vol. 36, no. 12, pp. 1415–1422, Oct. 2017.
[15] C. Brumann, M. Kukuk, and C. Reinsberger, "Evaluation of open-source and pre-trained deep convolutional neural networks suitable for player detection and motion analysis in squash," *Sensors*, vol. 21, no. 13, p. 4550, Jul. 2021, doi: 10.3390/s21134550.
[16] S. Katoch, S. S. Chauhan, and V. Kumar, "A review on genetic algorithm: Past, present, and future," *Multimedia Tools Appl.*, vol. 80, pp. 8091–8126, Oct. 2020.
[17] Y. Hong, P. D. Robinson, W. K. Chan, C. R. Clark, and T. Choi, "Notational analysis on game strategy used by the world's top male squash players in international competition," *Austral. J. Sci. Med. sport*, vol. 28, pp. 18–23, Mar. 1996.
[18] S. Barris and C. Button, "A review of vision-based motion analysis in sport," *Sports Med.*, vol. 38, no. 12, pp. 1025–1043, Dec. 2008.
[19] J. Pers, G. Vuckovic, S. Kovacic, and B. Dezman, "A low-cost real-time tracker of live sport events," in *Proc. 2nd Int. Symp. Image Signal Process. Anal., 23rd Int. Conf. Inf. Technol. Interfaces*, Jun. 2001, pp. 362–365.
[20] J. Perš and S. Kovačić, "Computer vision system for tracking players in sports games," in *Proc. 1st Int. Workshop Image Signal Process. Anal., 22nd Int. Conf. Inf. Technol. Interfaces*, Jun. 2000, pp. 177–182.
[21] G. Vučković, J. Perš, N. James, and M. Hughes, "Measurement error associated with the SAGIT/Squash computer tracking software," *Eur. J. Sport Sci.*, vol. 10, no. 2, pp. 129–140, Mar. 2010.
[22] R. Y. Tsai, "A versatile camera calibration technique for high-accuracy 3D machine vision metrology using off-the-shelf TV cameras and lenses," *IEEE J. Robot. Autom.*, vol. RA-3, no. 4, pp. 323–344, Aug. 1987.
[23] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 11, pp. 1330–1334, Nov. 2000.
[24] M. Kano, H. Okubo, M. Takahashi, K. Ikeya, K. Hisatomi, and T. Mishina, "Accurate and practical calibration of multiple pan-tilt-zoom cameras for live broadcasts," *IEEE Access*, vol. 8, pp. 153993–154006, 2020.
[25] Q. Ji and Y. Zhang, "Camera calibration with genetic algorithms," *IEEE Trans. Syst., Man, A, Syst. Humans*, vol. 31, no. 2, pp. 120–130, Mar. 2001.
[26] P. Liu, J. Zhang, and K. Guo, "A camera calibration method based on genetic algorithm," in *Proc. 7th Int. Conf. Intell. Hum.-Mach. Syst. Cybern.*, Aug. 2015, pp. 565–568.
[27] R. I. Hartley and A. Zisserman, *Multiple View Geometry in Computer Vision*, 2nd ed. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[28] D. C. Brown, "Decentering distortion of lenses," *Photogram. Eng.*, vol. 32, no. 3, pp. 444–462, 1966.
[29] G. Borgefors, "Distance transformations in digital images," *Comput. Vis., Graph., Image Process.*, vol. 34, no. 3, pp. 344–371, 1986.
[30] T. H. Rowan, "Functional stability analysis of numerical algorithms," Ph.D. dissertation, Dept. Comput. Sci., Univ. Texas Austin, Austin, TX, USA, 1990.
[31] J. Gablonsky and C. Kelley, "A locally-biased form of the direct algorithm," *J. Global Optim.*, vol. 21, no. 1, pp. 27–37, 2001.
[32] S. G. Johnson. (2011). *The NLopt Nonlinear-Optimization Package*. [Online]. Available: https://github.com/stevengj/nlopt
[33] B. O. Community. (2018). Blender—A 3D Modeling Rendering Package. Blender Foundation. Stichting Blender Foundation. Amsterdam, The Netherlands. [Online]. Available: http://www.blender.org
[34] J. Canny, "A computational approach to edge detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. PAMI-8, no. 6, pp. 679–698, Nov. 1986.

C. BRUMANN received the B.Sc. and M.Sc. degrees in computer science from the Dortmund University of Applied Sciences and Arts, Dortmund, Germany. He is currently pursuing the Ph.D. degree with the Institute of Sports Medicine, Paderborn University, Paderborn, Germany. Since 2017, he has been with the Department of Computer Science, Dortmund University of Applied Sciences and Arts. His research interests include signal/image processing and video understanding with an emphasis on sports and health related applications.

M. KUKUK received the Diploma degree *(summa cum laude)* in computer science and the doctorate degree in computer science from the University of Dortmund, Dortmund, Germany. He is currently pursuing the Ph.D. degree with the Medical Imaging Department, Siemens Corporate Research, Princeton, NJ, USA. He received the Doctorate Scholarship from the German Academic Exchange Service. He was a Research Scientist at Siemens Medical Solutions, Stanford University School of Medicine, California, and a Senior Research Scientist at Fujifilm Medical Systems, San Jose, California. He is currently a Professor of computer science and medical engineering with the Dortmund University of Applied Sciences and Arts, Dortmund. His research interests include biomedical signal/image processing and analysis and machine learning.