Hindawi
Advances in Multimedia
Volume 2022, Article ID 2743696, 12 pages
https://doi.org/10.1155/2022/2743696

*Research Article*

# Research on Moving Target Detection and Tracking Technology in Sports Video Based on SIFT Algorithm

**Zhu Mei and Yue Wang**

*Shenyang Jianzhu University, Shenyang 110168, China*

Correspondence should be addressed to Zhu Mei; meizhu@sjzu.edu.cn

Received 27 December 2021; Revised 16 January 2022; Accepted 21 January 2022; Published 11 February 2022

Academic Editor: Qiangyi Li

Sports video moving target detection and tracking play an important role in enhancing the popularity of sports and the promotion of sports events. This paper combines the SIFT algorithm to carry out the research of sports video moving target detection and tracking technology, to identify sports features, and to improve the sports feature detection algorithm. Moreover, this paper divides the point cloud data into multiple cube grids under the coordinate system where it is located, and then finds the center of gravity of the data points in each grid, and replaces the coordinates of all points in the grid with the coordinates of the center of gravity. In addition, this paper combines data analysis to verify the algorithm and build a sports video moving target detection system. The experimental research results verify that the sports video target detection and tracking technology based on the SIFT algorithm proposed in this paper has good results.

## 1. Introduction

In recent years, sports behavior recognition technology has been increasingly integrated into daily sports video analysis. Moreover, there are more and more scientific researchers engaged in sports behavior recognition, and the research on behavior recognition technology is in full swing. In addition, the emergence of various latest methods and theories and the introduction of many new algorithms from other fields have made great progress in behavior recognition [1]. The main process of sports behavior recognition technology can be roughly divided into these four steps: feature extraction, feature representation, behavior modeling, and behavior classification [2]. According to specific research goals and needs, corresponding changes can be taken in these steps. For example, some algorithms directly combine feature extraction and representation into one step, and some methods do not even require behavior modeling to send the descriptors after feature extraction and representation directly into the classifier to recognize their classification. At the same time, some methods incorporate iterative feedback processes such as deep learning. In addition, some methods also perform further processing (such as dimensionality reduction, etc.) on the descriptors after the feature representation to make these features more distinguishable, etc. [3].

The models used in time series modeling can be regarded as further expressions after feature representation. The time series information extracted by these models is not presented in front of people in an intuitive form but through the parameters of the model after modeling. Make an expression. Existing methods for time series modeling include hidden Markov models, conditional random fields, linear dynamic systems, and the recently popular recurrent neural network models.

This article combines the SIFT algorithm to carry out the research of sports video moving target detection and tracking to identify sports features, which provides a theoretical reference for the application of dynamic recognition technology in sports competitions and sports training.

## 2. Related Work

The visual SLAM system completes the estimation of its position in the environment (white positioning), mainly relying on the visual odometer module [4]. The binocular visual odometer calculates the depth information by calculating the parallax between the left and right cameras, while the monocular visual odometer cannot obtain absolute scale information and can only be obtained through other sensors or environmental information [5]. Although the current advanced visual odometer has been able to run in real time and obtain high positioning accuracy, almost all methods are assumed to operate in a static environment [6]. If there is interference from moving objects in the field of view of the sports camera, the visual odometer will have a large estimation error or even fail. Aiming at the problem of moving objects in the scene interfering with the visual odometer, the random sampling consensus algorithm is currently the most mature and effective method [7]. This method fits the model and eliminates the data points that are inconsistent with the model as outliers. When the moving object in the camera's field of view only occupies a small part, the RACSAC method can be used to filter the feature points, and the feature points on the moving object can be removed as external points, and combined with the motion estimation in the visual odometer, thus get better positioning results [8]. But when the moving object occupies a large part of the camera's field of view, the RANSAC algorithm may also treat the characteristic points on the moving object as interior points. At this time, relying on this method will not be able to achieve the purpose of eliminating interference [9]. Literature [10] uses pretraining to segment the feature points in the image into dynamic and static feature points, but this method is difficult to implement in actual real-time applications. Literature [11] relies on the dense scene flow to segment dynamic objects, but the scene flow calculation also needs to use the visual mileage calculation method to compensate for the posture. Literature [12] uses image segmentation technology to segment the image into static background and motion regions and uses the partitioned regions to perform motion estimation separately and then perform a global fusion. Although this algorithm has high accuracy, it is difficult to meet real-time requirements. Literature [13] uses the IMU information as a priori to segment the dynamic feature points in the image and realize the visual positioning of the dynamic environment by combining the information of the inertial navigation system and the depth visual odometer.

Literature [14] uses a single-chip microcomputer as the controller to design a high-speed positioning control system for image monitoring dynamic brackets and dynamically calculates the rotation angle of the pan/tilt based on the control motor; Literature [15] analyzes the sports trajectory tracking strategy to design a dynamic Fuzzy PID controller for point-line calculation trajectory tracking, research on the visual pan-tilt pose calculation and dynamic trajectory tracking system control technology.

The vigorous development of machine vision under the compatibility of various devices also marks the beginning of vision requirements. Literature [16] analyzes the research on the coordination strategy of the multimovement sports system to capture the moving target and uses the stereo vision motion detection to complete the motion parameter estimation of the moving target. At present, the three-dimensional object positioning technology based on binocular stereo vision has become one of the hot spots in the research of vision measurement [17]. Compared with the monocular camera motion measurement technology in the large tank discharge hole visual positioning control system, stereo vision can calculate more information when the parallax is obtained. It is not only compatible with the characteristics of the monocular camera but also can be used to construct three-dimensional objects. The accuracy and applicability of the model are high [18].

## 3. Moving Target Detection Based on SIFT Algorithm

This article analyzes the moving target detection algorithm, and this part mainly combines the SIFT algorithm to identify and track the moving target.

The essence of the Nonmaximum Suppression (NMS) algorithm is to retain the most significant or least significant key points within a certain range from the key points initially extracted by the algorithm and discard other key points.

The specific steps are as follows.

For a point $kp_i$ in the key point set $KP$, (1) the algorithm first takes its neighborhood $\mathrm{nbhd}(kp_i)$ and judges whether its saliency value is the largest or smallest in the neighborhood. If the maximum or minimum value is taken, it is marked as the true key point; otherwise, it is marked as the rest key point. (2) The algorithm traverses the entire set of key points $KP$ and removes all the key points marked as surplus points.

The saliency in the above process can be determined according to the algorithm or requirements, and features such as curvature, the interval length of the normal vector distribution of the neighborhood points, and the shape index value can also be used.

According to the above method, if the neighborhood of size $k = 3$ is taken, it is determined whether the curvature of each key point in the graph is the maximum value in the neighborhood, and the nonmaximum points are removed, as shown in Figure 1. This example uses curvature as the saliency, which not only retains the high-quality key points but also greatly reduces the redundancy.

The Intrinsic Shape Signatures (ISS) algorithm was proposed by Zhong et al. The ISS algorithm uses the dispersion difference degree between the three main directions of the $p$ point neighborhood $\mathrm{nbhd}(p)$ local reference coordinate system as the evaluation index of the significance of the $p$ point and extracts the points with a large dispersion difference degree by comparing with the preset threshold.

First, we use the PCA algorithm to calculate the three eigenvalues $\lambda_1, \lambda_2, \lambda_3$ of the $\mathrm{nbhd}(p)$ covariance matrix $\mathrm{cov}(p)$ in descending order. The three eigenvalues obtained after the eigenvalue decomposition of the covariance matrix of $\mathrm{nbhd}(p)$, respectively, represent the degree of dispersion along the three eigendirections $v_1$, $v_2$, and $v_3$. Therefore, the ratio of the two eigenvalues can be used to express the degree of difference in the dispersion of the two main axis directions.

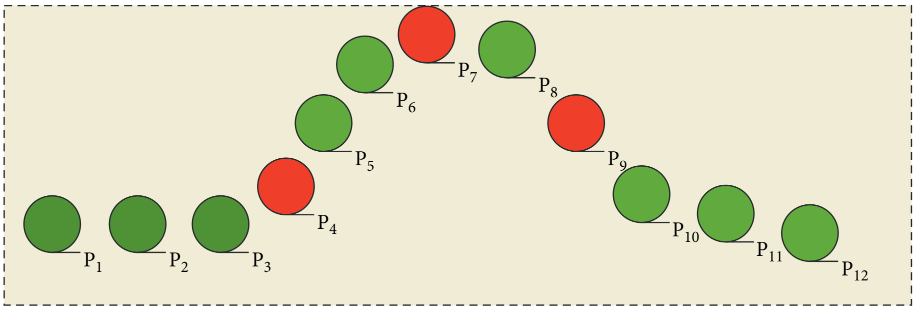

FIGURE 1: Key points after nonmaximum suppression.

$$\frac{\lambda_2}{\lambda_1} < Th_{12} \cap \frac{\lambda_3}{\lambda_2} < Th_{23}. \quad (1)$$

Formula (1) shows the criterion for extracting key points, where $Th_{12}$ and $Th_{23}$ are preset thresholds, and the size of the threshold determines how many key points are extracted. The smaller the threshold value is, the more key points are extracted.

For any point $p_i$ in the point cloud, according to formula (2), it calculates the centroid $p_i$ of the neighborhood $\text{nbhd}(p_i)$ of the $p_i$ point and transforms it to the centroid system, calculates the covariance matrix $\text{cov}(p_i)$, and then decomposes the eigenvalue of the covariance matrix $\text{cov}(p_i)$, as shown in the following formula [19]:

$$\text{cov}V = \Lambda V. \quad (2)$$

Then, the Hotelling transform is performed on the neighborhood point $q_j \in \text{nbhd}(p_i)$, and the neighborhood coordinates of the point $p_i$ are projected onto the three principal axes, as shown in the following formula:

$$q_j' = V\left(q_j - p_c\right). \quad (3)$$

$q_j$ is the coordinate of the neighborhood point before transformation and $q_j$ is the coordinate after transformation. Then, the algorithm calculates the ratio $\delta$ between the *nbh* $d(p_i)$ coordinate distribution ranges on the $x$ and $y$ axes (the first and second largest spindles), as shown in formula (4), where $X = \left\{x_{qj}, q_j \in \text{nbhd}(p_i)\right\}, Y = \left\{y_{qj}, q_j \in \text{nbhd}(p_i)\right\}$ [20].

$$\delta = \frac{\max(X) - \min(X)}{\max(Y) - \min(Y)}. \quad (4)$$

For symmetrical surfaces (that is, the neighbors are distributed in the same direction along the largest and second largest axes), the value of $\delta$ is equal to 1; for asymmetric surfaces, $\delta$ is greater than 1. The algorithm sets the threshold as $th_1$ $(t_1 > 1)$and determines whether $\delta > th_1$ is satisfied, and if it is satisfied, it is recorded as a key point.

The algorithm traverses each point in the point cloud $P$ and completes the preliminary basket selection of the key points, which is marked as $KP$.

For the key point $kp_i$ that has been preliminarily selected, a quadric surface fitting is performed on its neighborhood to obtain a parametric surface S. Subsequently, a uniform $n \times n$ grid $(n = 20)$ is used to sample the fitted surface, and the principal curvature $k_1, k_2$ and Gaussian curvature $K$ at the sampling point are calculated, and the parameter $d$ is used to evaluate the quality of key points according to formulas (5) and (6). In this paper, in order to simplify the calculation, the algorithm uses neighborhood points instead of uniform sampling points to calculate $Q_k$, and $n$ is the number of neighborhood points [21].

$$K = k_1 k_2, \quad (5)$$

$$\begin{aligned} Q_k = \frac{1000}{n^2} &\sum |K| + \max(100K) + |\min(100K)| \\ &+ \max(10k_1) + \min(10k_2). \end{aligned} \quad (6)$$

Finally, $Q_k$ is the saliency parameter. This article uses the NMS algorithm to perform nonmaximum suppression, where only the maximum value is retained to complete the screening of key points.

The Local Surface Patch (LSP) algorithm uses the least squares method to fit a local point cloud into a parametric surface. It calculates the first basic quantity and the second basic quantity of the surface and then constructs the shape index $Si(p_i)$ and filters out the points whose $Si(p_i)$ satisfies certain conditions as the key points. Finally, the NMS algorithm with $Si$ as the saliency parameter, further filters the initially selected key points to complete the final keypoint detection. The specific process is as follows.

For the neighborhood $\text{nbhd}(p_i)$ of any point $p_i$ in the point cloud $P$, the algorithm first establishes the LRF and rotates the neighborhood $\text{nbhd}(p_i)$ to the three principal axis directions of the LRF to eliminate the influence of the initial pose on further calculations. Then, the quadric surface $s(p_i)$ is fitted, and the principal curvature $k_1, k_2$ at the point $p_i$ on the surface $s(p_i)$ is calculated according to the following formula [22]:

$$Si(p_i) = \frac{1}{2} - \frac{1}{\pi}\tan^{-1}\frac{k_1 + k_2}{k_1 - k_2}. \quad (7)$$

It can be seen that the value range of the shape index value $Si$ defined by the above formula is [0,1]. When the $Si$ value is large, the corresponding partial surface is convex; when the $Si$ value is small, the corresponding partial surface is concave.

After completing the $Si$ calculation of all points in the point cloud, the preliminary screening of the key points can be completed according to the following formula:

$$\begin{aligned} &Si(p_i) \geq (1+\alpha)\mu_{Si(p_i)}, \\ &\text{or } Si(p_i) \leq (1-\beta)\mu_{Si(p_i)}. \end{aligned} \tag{8}$$

Among them, $\mu_{Si(p_i)}$ is the mean value of $Si$ in the neighborhood of $p_i$, $\alpha$ and $\beta$ are preset parameters, and the value of $\alpha$ and $\beta$ should be between 0 and 1.

$$Si(p_i) = \frac{1}{n} \sum_{p_j \in \mathrm{nbhd}(p_i)} Si(p_j). \tag{9}$$

Then, $Si$ is used as a saliency parameter. Using the NMS algorithm, this paper judges whether the $Si$ value is the maximum or minimum value of the $Si$ value of each point in the neighborhood point by point. If it is, keep it. If it is not, delete it. Finally, the key point set $KP$ is obtained.

The Histogram of Normal Orientation (HoNO) algorithm first calculates the angle between the normal vector of each point in the neighborhood $\mathrm{nbhd}(p_i)$ of each point $p_i$ in the point cloud $P$ and the normal vector of the target point $p_i$, and then counts them to form a histogram. According to the histogram characteristics, the flat area is excluded and the salient area of the feature is detected. Then, by evaluating the properties of the histogram and the neighborhood covariance matrix, key points are extracted from the salient regions.

First, for each point $p_i \in P$, this paper uses the PCA algorithm to estimate the normal vector $n_{i^o}$. Then, for every point $q_j \in \mathrm{nbhd}(p_i)$ except $p_i$ in the neighborhood $\mathrm{nbhd}(p_i)$ of $p_i$, calculate the normal vector angle $\langle p_i, q_j \rangle$ between $q_j$ and $q_i$ according to formula (10) and count the included angles into the histogram $H_i$ containing $N$ boxes, where the length of each box is 10 degrees. In order to eliminate the influence of the neighborhood point density on the algorithm, it is necessary to normalize the histogram after the large-angle filling angle of the normal vectors of all neighborhood points is completed.

$$\theta_{\mathrm{deg}} = \left( \tan^{-1} \frac{\left\| N_i \times N_j \right\|}{N_i \cdot N_j} \right) \frac{180}{\pi}. \tag{10}$$

Obviously, the histogram $H_i$ of the point $p_i$ with the neighborhood approximate to the plane has the characteristic of "the first box has a higher value and the rest of the box values are approximately 0." Similarly, the area with a larger degree of curvature has a large normal vector distribution range. Therefore, most of the box values in its histogram are nonzero. Therefore, it is necessary to design parameters to describe the distribution of values in the histogram.

$$\mathrm{Kurt}(H) = \frac{\sum_{k=1}^{N} \left(H_k - \overline{H}\right)^4}{NS^4} - 3, \tag{11}$$

$$S = \sqrt{\frac{\sum_{k=1}^{N} \left(H_k - \overline{H}\right)^2}{N}}. \tag{12}$$

As shown in formula (11), Kurt is used to express the kurtosis and dispersion of the histogram distribution. If the parameter kurtosis parameter $\mathrm{Kurt}(H_i)$ of the point $p_i$ histogram is less than the preset parameter $Th$, there is no obvious presence in the histogram. That is, the distribution range of the values in the histogram is larger, and $p_i$ is retained as the key point; otherwise, it is removed.

Finally, after the Kurt parameter calculation of all points and the determination of the preliminary key points have been completed, the key point set is deredundant by using $\mathrm{Kurt}(H)$ as the saliency parameter and NMS is used to obtain the final key point set $KP$.

The Harris operator is extended to three-dimensional space, and the specific steps are as follows.

First, for the point $p_i$ in the point cloud $P$, the algorithm queries the neighborhood $\mathrm{nbhd}(p_i)$ and establishes the LRF, and the neighborhood $\mathrm{nbhd}(pi)$ is translated to the LRF coordinate system that is the origin of $p_i$.

After establishing LRF, the algorithm sets the parameters according to formula (13) and performs quadric surface fitting on the neighborhood to obtain the fitted quadric surface parameter $p_1, p_2, \ldots, p_6$.

$$\begin{aligned} z &= f(x,y) \\ &= \frac{p_1}{2}x^2 + p_2 xy + \frac{p_3}{2}y^2 + p_4 x + p_5 y + p_6. \end{aligned} \tag{13}$$

For parametric surfaces, adding more high-order terms means that it can adapt to more complex surfaces. However, more complex surfaces do not have clearly defined derivatives at certain points in the defined domain. Moreover, when the neighborhood radius is not large, the vicinity of the target point can be approximated as a quadric surface. Therefore, the directional derivative can be easily obtained according to the following equations:

$$f_x = \frac{\partial f(x,y)}{\partial x}|_{x=0}, \tag{14}$$

$$f_y = \frac{\partial f(x,y)}{\partial y}|_{y=0}. \tag{15}$$

Considering the influence of noise, the Gaussian function originally proposed by Harris and Stephens can be applied, as shown in the following equations:

$$A = \frac{1}{\sqrt{2\pi}\sigma} \int_0^{R^2} e^{-\left(x^2+y^2\right)/2\sigma^2} f_x(x,y)^2 \mathrm{d}x\mathrm{d}y, \tag{16}$$

$$B = \frac{1}{\sqrt{2\pi}\sigma} \int_0^{R^2} e^{-\left(x^2+y^2\right)/2\sigma^2} f_x(x,y)^2 \mathrm{d}x\mathrm{d}y, \tag{17}$$

$$C = \frac{1}{\sqrt{2\pi}\sigma} \int_0^{R^2} e^{-\left(x^2+y^2\right)/2\sigma^2} f_x(x,y) f_y(x,y) \mathrm{d}x\mathrm{d}y. \tag{18}$$

Substituting the quadric surface equation, it is simplified as shown in the following equations:

$$A = p_4^2 + 2p_1^2 + 2p_2^2, \tag{19}$$

$$B = p_5^2 + 2p_2^2 + 2p_3^2, \tag{20}$$

$$C = p_4 p_5 + 2p_1 p_2 + 2p_2 p_3. \tag{21}$$

Then, the analysis matrix $E$ is constructed as follows:

$$E = \begin{pmatrix} A & C \\ C & B \end{pmatrix}. \tag{22}$$

By analyzing the determinant value and trace of the matrix $E$, the Harris corner response value $h(p_i)$ at the point $p_i$ is constructed. As shown in formula (23), $k$ is a non-negative preset parameter.

$$h(p_i) = \det(E) - k(tr(E))^2. \tag{23}$$

The specific steps to implement the 3D-SIFT algorithm in this paper are as follows:

(1) The algorithm constructs the point cloud scale space in a three-dimensional coordinate system.

The scale space of the point cloud is $L(x, y, z, \sigma)$, the point whose coordinate is $(x, y, z)$ and the pixel value $I(x, y, z)$ in its neighborhood are convolved with a three-dimensional Gaussian function $G(x, y, z, \sigma)$ whose scale can be changed by changing $\sigma$, as shown in the following formula:

$$L(x, y, z, \sigma) = G(x, y, z, \sigma) * I(x, y, z). \tag{24}$$

Among them, the specific expression of the Gaussian function is shown in the following formula:

$$G(x, y, z, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left( -\frac{\left(x^2 + y^2 + z^2\right)}{2\sigma^2} \right). \tag{25}$$

The definition of the cube grid size during downsampling is shown in the following formula:

$$2^k \sigma, \quad k = 0, 1, \ldots. \tag{26}$$

(2) The algorithm builds the DoG space of the 3D point cloud.

The process of downsampling is equivalent to performing local mean filtering on the point cloud set. The local features in each cube grid disappear, replaced by the center of gravity coordinates, and there will be discontinuities between the grids. Therefore, in order to ensure that the algorithm can find characteristic points in the point cloud stably, it is necessary to construct a DoG space for the point cloud. The construction formula is shown in the following formula:

$$\begin{aligned} D(x, y, z, \sigma) &= [G(x, y, z, k\sigma) - G(x, y, z, k\sigma)]^* I(x, y, z) \\ &= L(x, y, z, k\sigma) - L(x, y, z, k\sigma). \end{aligned} \tag{27}$$

The construction of the DoG space of 3D point cloud data is simply the process of convolving the Gaussian scale function $G(x, y, z, \sigma)$ with the coordinate data $I(x, y, z)$ of the point. It is equivalent to smoothing the point cloud data layer by layer, and each layer needs to be divided into several small scales distinguished by a certain step length, as shown in formula (28). In the formula, $T$ is a preset parameter for calculating the Gaussian scale space.

$$\sigma' = \sigma \cdot 2^{t/T}, \quad t = 0, 1, \ldots. \tag{28}$$

(3) The algorithm calculates the Gaussian filter response value of the sampling point in the Gaussian scale space.

When calculating the Gaussian filter response value, in order to improve the efficiency of data processing, the effect of increasing the distance between the points on the characteristics of the sampling points can be considered (here, the curvature $c$ is used as the main geometric feature). It is considered that the closer the point to the sampling point $p_{0i}$ is, the greater the contribution to its curvature. Therefore, it is possible to consider weighting the coefficient $w_j$ of the distance dist $< 3\sigma$ from the point $p_{0i}$. Then, the Gaussian filter response value $F$ can be calculated according to the following formula:

$$F = \frac{\sum c_j w_j}{\sum w_j}. \tag{29}$$

In formula (29), $c_j$ is the curvature value of the neighboring point $p_{oj}$ of the sampling point $p_{oi}$. The calculation formula of the weighting coefficient $w_j$ is shown in formula (30). In the formula, $\text{dist}^2(p_{0i}, p_{0j})$ represents the square of the distance between the sampling point $p_{0i}$ and the neighboring point $p_{0j}$.

$$w_j = e^{-\text{dist}^2\left(p_{0i}, p_{0j}\right)/2\sigma^2}. \tag{30}$$

According to the above formula, the corresponding value of Gaussian filtering for each data point in the 3D point cloud data can be calculated. By calculating the difference between the Gaussian filter response value $F$ of the sampling point at the current scale and the Gaussian filter response value $F$_last at the previous scale, the DoG value of the sampling point at the current scale can be obtained, as shown in the following formula:

$$\text{DoG}\left(p_i, \sigma - 1\right) = F - F_{\text{last}}. \tag{31}$$

The algorithm repeats the above steps for all points in the point cloud at each scale until all the points in the point cloud are traversed to obtain all the DoG values of all points in the point cloud at different scales.

(4) The algorithm detects extreme points in the DoG space of the point cloud.

In the DoG space of point cloud of various scales, respectively, the DoG extremum of the neighboring point is found in the neighborhood of the current point. If the DoG value under a certain scale is greater than the DoG value under two adjacent scales, the current point under this scale is the key point. The feature points found by this method are scale-invariant and can be retained as feature points in different scale spaces.

## 4. Research on Sports Video Moving Target Detection and Tracking Based on SIFT Algorithm

The data set of this paper mainly comes from the network, and the data set of this paper is shown in Figure 2.

The key point detection experiment in this paper is based on the model point cloud of the above data set. The evaluation index is calculated in each data set and the average value is calculated. Finally, the average value of the data set is calculated and the corresponding curve is drawn.

This paper chooses the relative repetition rate, the accuracy rate of the descriptor matching experiment, and the operation efficiency as the evaluation indicators.

(1) Relative repeatability (repetition rate). The repetition rate represents the consistency between the key points detected when the point cloud $P$ changes and the key points detected by the point cloud before the change. As shown in formula (32), $KP$ is the set of key points detected by the point cloud $P$, $KP\prime$ is the set of key points detected by the changed point cloud $P'$, then $R_{KP}$ represents the same part in $KP'$ and $KP$.

$$r = \frac{\text{length}\left(R_{KP}\right)}{\text{length}\,(KP)}. \tag{32}$$

The repetition rate is used to evaluate the robustness of the algorithm to factors such as spatial transformation, noise, and resolution. The noise repetition rate $r_{\text{noise}}$ represents the proportion of the same part of the key points detected before and after adding noise. The outlier repetition rate $r_{\text{outlier}}$ represents the proportion of the same part of the key points detected before and after adding the outlier. The grid resolution repetition rate $r_{\text{density}}$ represents the proportion of the same part of the key points detected before and after the point cloud sampling.

In order to quantitatively set the neighborhood radius $r$, for each data set, the algorithm first calculates the diagonal length $r_i$ of the spatial bounding box of the data set model point cloud $P_i$ and traverses all models to obtain the maximum length $r_{\max}$. Then, the algorithm reduces all the point coordinates in the point cloud by $r_{\max}$ times, and in the subsequent evaluation experiment, the neighborhood radius $r$ is uniformly set to 0.02.

The evaluation experiment includes three parts: repetition rate experiment, running time experiment, and descriptor matching experiment. Algorithms and test programs are written in MATLAB language.

The repetition rate experiment mainly includes two parts: the parameter change module and the repetition rate calculation module.

(1) The point cloud quality change module has three modes to choose from: adding Gaussian noise, changing the point cloud density, and adding outliner's. They correspond to the repeatability of the key point detection algorithm to noise, the repeatability of the grid resolution, and the repeatability of the outliers.

In order to quantitatively obtain the amplitude of the noise signal conforming to the Gaussian distribution, this paper sets the parameter $k$ and combines the neighborhood radius $r$ used by the key point detection algorithm, sets $kr$ as the maximum amplitude $N$ of Gaussian noise, and calculates noise amplitude distribution by formula (33). Among them, $\text{rand}\,(a, b)$ represents a random number with a value range of $[a,\ b]$.

$$n = N\sqrt{-2\,\text{logrand}\,(0,1)}\,\cos\left(2\pi\text{rand}\,(0,1)\right). \tag{33}$$

After the algorithm calculates the noise amplitude $n$, it chooses random direction angles $\theta$ and $\varphi\,(\theta - \text{rand}\,(0, 2\pi)),\ \varphi = \text{rand}\,(0, \pi/2)$ and calculates the unit vector $l$ according to formula (34). Finally, the algorithm calculates the coordinate $\hat{p}' = p + n l^{\circ}$ after noise is added, $p$ is the point cloud coordinate before noise is added, and $p'$ is the point cloud coordinate after noise is added.

$$l = (\cos(\theta)\sin(\varphi),\ \sin(\theta)\sin(\varphi),\ \cos(\varphi)). \tag{34}$$

Outliers are a common type of noise in the point cloud obtained by three-dimensional scanning, which manifests as noise points far away from the surface of the object. Similarly, the algorithm uses random sampling to select a certain proportion of points $p_o$ in the point cloud $P$ as outliers. For point $p_o$, the algorithm first calculates the normal vector $n_o$, increases the coordinate value along the direction of the normal vector $n_o$, and sets the increment size to 1 times the radius of the neighborhood. As shown in formula (35), the outlier set $p_o'$ is obtained. In order to observe more intuitively, it is displayed in the form of a patch.

$$p_o' = p_0 + r n_o. \tag{35}$$

In addition, in the repetitive experiments of the key point algorithm on the space transformation, the point cloud needs to be spatially transformed. Space transformation can be realized by rotation and translation, as shown in equation (36). By realizing the rotation and translation of all the points in the point cloud $P$, the transformed point cloud $P'$ is obtained.

$$P' = \left(R \cdot P^{\text{Trans}} + T\right)^{\text{Trans}}. \tag{36}$$

Among them, $P^{\text{Trans}}$ represents the transpose of $P$, and the rotation matrix $R$ and translation matrix $T$ are obtained by formulas (37) and (38), respectively.

Figure 2: Sports data set.

$$R = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos(\alpha) & -\sin(\alpha) \\ 0 & \sin(\alpha) & \cos(\alpha) \end{bmatrix} \cdot \begin{bmatrix} \cos(\beta) & 0 & \sin(\beta) \\ 0 & 1 & 0 \\ -\sin(\beta) & 0 & \cos(\alpha) \end{bmatrix} \cdot \begin{bmatrix} \cos(\beta) & -\sin(\gamma) & 0 \\ \sin(\gamma) & \cos(\beta) & 0 \\ 0 & 0 & 1 \end{bmatrix}, \quad (37)$$

$$T = \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix}. \quad (38)$$

(2) After completing the detection of the key point KP of the original point cloud and the key point KP′ of the transformed point cloud, it is necessary to calculate the same part of the two key point sets, that is, the repetitive calculation module.

Figure 3 of the repetition rate experiment shows that the algorithm selects a data set. In the model of the data set, the repetition rate change of the selected key point algorithm is tested when the selected conditions change. We use the noise repetition rate $r_{\text{noise}}$ as an example. We must first add noise to the model point cloud, and then use the key point detection algorithm tested to perform key point detection on the point cloud before and after adding noise, and finally use Algorithm 1 or Algorithm 2 to calculate the repetition rate of the two sets of key points.

Descriptor matching experiments need to be combined with the PRC drawing process in the evaluation of the description ability of feature descriptors. The calculation process of PRC covers the complete target recognition process including feature matching, verification, and other steps. After selecting different key point detection algorithms and combining them with the same descriptor, the higher the accuracy of feature matching, the higher the quality of the extracted key points. First, we need to detect the key points of the scene point cloud $P_{\text{scene}}$ and the model point cloud $P_{\text{model}}$ , respectively, and establish a feature descriptor $F_{\text{scene}}, F_{\text{model}}$. Then, we establish the corresponding relationship between the scene feature and the model feature and calculate the matching accuracy rate. According to the change trend of the PRC curve, we can determine the descriptive strength. That is, we know the effect of the key point detection algorithm and the feature descriptor.

## 5. Experimental Results and Analysis

*5.1. Spatial Transformation Repetition Rate $r_{trans}$.* The original point cloud is rotated along the three coordinate axes of $x, y, z$ by $\pi/8, \pi/4, \pi/3, \pi/2, \pi 3\pi/2, 2\pi$, the key points before and after the rotation are detected, and the repetition rate is calculated. The repetition rate calculated on the 4 data sets is shown in Table 1. The rotation angle-repetition rate curve shown in Figure 4 is drawn based on the average value of $r_{\text{noise}}$ on 4 data sets. Table 1 is the average value of the spatial

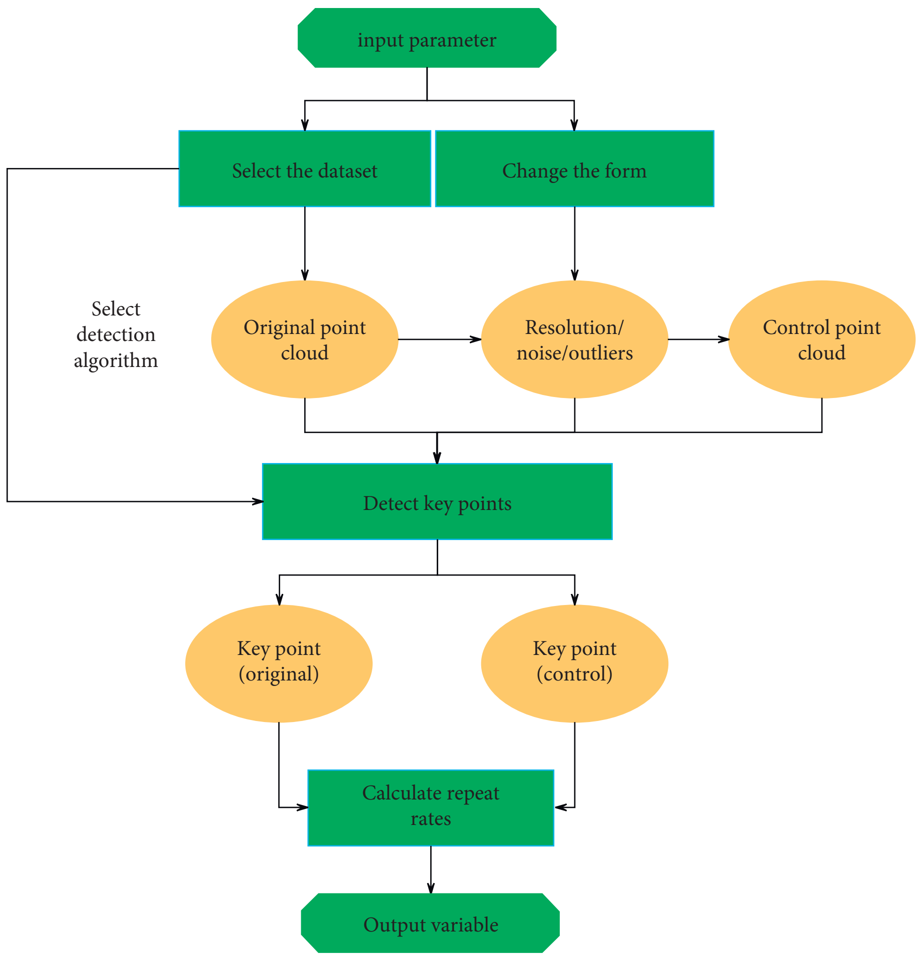

FIGURE 3: Flow chart of repetition rate experiment.

TABLE 1: The average value of the spatial repetition rate $r_{\text{noise}}$ of the key point detection algorithm on the 4 data sets.

| Rotation angle ($\pi$) | 1/8 | 1/4 | 1/3 | 1/2 | 1 | 3/2 | 2 |
|---|---|---|---|---|---|---|---|
| ISS | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| KPQ | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| HoNO | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| LSP | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 3D-Harris | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |
| 3D-SIFT | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 | 1.0 |

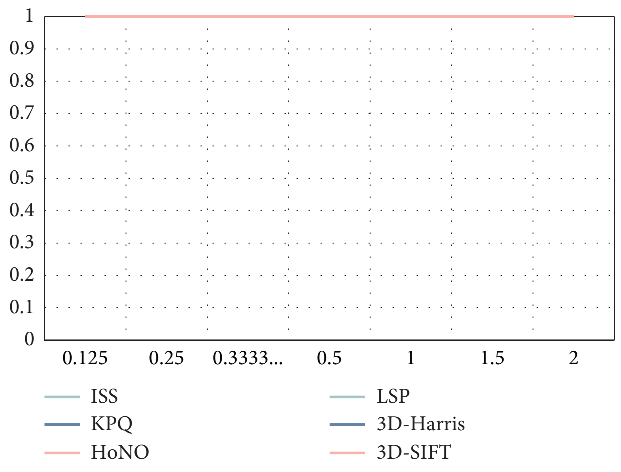

FIGURE 4: Rotation angle-repetition rate change curve.

repetition rate $r_{\text{trans}}$ of the key point detection algorithm on the 4 data sets.

The six algorithms tested have a repetition rate of 1 in all tests. That is, they have spatial transformation invariance. Some of these six algorithms realize the invariance of space transformation, and some of them are described in the next step by selecting features that have nothing to do with the choice of coordinate system. For example, the ISS algorithm uses the three eigenvalues of the covariance matrix, 3D-SIFT uses the layer constructed by curvature, and the normal vector angle is used by HoNO. Others describe features directly in the local reference coordinate system. For example, LSP and 3D-Harris both fit the quadric surface in the local reference coordinate system.

*5.2. Gaussian Noise Repetition Rate $r_{noise}$.* The original point cloud adds Gaussian noise according to the maximum noise amplitude of $0.05r$, $0.08r$, $0.1r$, $0.15r$, and $0.2r$. The key points before and after adding the noise are detected, and the repetition rate is calculated. The $r_{\text{noise}}$ on the 4 data sets are shown in Tables 2–5. The Gaussian noise-repetition rate

Table 2: Gaussian noise repetition rate $r_{\text{noise}}$ of the key point detection algorithm on the Laser Scanner data set.

| Peak intensity ($r$) | 0.05 | 0.08 | 0.1 | 0.15 | 0.2 |
|---|---|---|---|---|---|
| ISS | 0.660944 | 0.511464 | 0.458439 | 0.352894 | 0.408949 |
| KPQ | 0.568529 | 0.46561 | 0.430967 | 0.341683 | 0.261388 |
| HoNO | 0.769721 | 0.77063 | 0.774064 | 0.772145 | 0.78376 |
| LSP | 0.407939 | 0.401172 | 0.389355 | 0.40097 | 0.390062 |
| 3D-Harris | 0.784265 | 0.702556 | 0.673367 | 0.497627 | 0.42117 |
| 3D-SIFT | 0.783053 | 0.708818 | 0.690234 | 0.585093 | 0.535502 |

Table 3: Gaussian noise repetition rate $r_{\text{noise}}$ of the key point detection algorithm on the Random View data set.

| Peak intensity ($r$) | 0.05 | 0.08 | 0.1 | 0.15 | 0.2 |
|---|---|---|---|---|---|
| ISS | 0.616302 | 0.580447 | 0.538229 | 0.445208 | 0.392183 |
| KPQ | 0.493082 | 0.455712 | 0.428341 | 0.361277 | 0.330169 |
| HoNO | 0.351076 | 0.304515 | 0.291385 | 0.275225 | 0.254924 |
| LSP | 0.205939 | 0.198162 | 0.19897 | 0.203515 | 0.230078 |
| 3D-Harris | 0.651046 | 0.614888 | 0.597213 | 0.531462 | 0.486113 |
| 3D-SIFT | 0.750632 | 0.702556 | 0.657914 | 0.574488 | 0.524594 |

Table 4: Gaussian noise repetition rate $r_{\text{noise}}$ of key point detection algorithm on Space Time data set.

| Peak intensity ($r$) | 0.05 | 0.08 | 0.1 | 0.15 | 0.2 |
|---|---|---|---|---|---|
| ISS | 0.647107 | 0.510757 | 0.464398 | 0.426018 | 0.396627 |
| KPQ | 0.625291 | 0.557722 | 0.511464 | 0.434199 | 0.365014 |
| HoNO | 0.603374 | 0.561762 | 0.543784 | 0.495809 | 0.49894 |
| LSP | 0.410363 | 0.414908 | 0.419251 | 0.414201 | 0.416423 |
| 3D-Harris | 0.837492 | 0.765378 | 0.728917 | 0.614585 | 0.534189 |
| 3D-SIFT | 0.829917 | 0.773862 | 0.707606 | 0.631452 | 0.57267 |

Table 5: Gaussian noise repetition rate $r_{\text{noise}}$ of the key point detection algorithm on the Kinect data set.

| Peak intensity ($r$) | 0.05 | 0.08 | 0.1 | 0.15 | 0.2 |
|---|---|---|---|---|---|
| ISS | 0.634583 | 0.488436 | 0.444703 | 0.395516 | 0.375114 |
| KPQ | 0.552672 | 0.479245 | 0.462075 | 0.399758 | 0.34643 |
| HoNO | 0.559843 | 0.502576 | 0.463186 | 0.419958 | 0.379154 |
| LSP | 0.399253 | 0.395314 | 0.37875 | 0.387335 | 0.389759 |
| 3D-Harris | 0.790123 | 0.745885 | 0.680235 | 0.57873 | 0.477629 |
| 3D-SIFT | 0.803859 | 0.724776 | 0.653167 | 0.543279 | 0.480962 |

variation curve shown in Figure 5 is drawn based on the average value of $r_{\text{noise}}$ on 4 data sets.

From the above experimental results, it can be seen that 3D-Harris uses the parameter characteristics of local quadric surface fitting, and 3D-SIFT uses the parameter characteristics of the covariance matrix. Both methods have a certain inhibitory effect on Gaussian noise. The ISS algorithm also uses the eigenvalues of the covariance matrix to determine the key points, but it does not use the scale space to further smooth it like 3D-SIFT, so it is more sensitive to noise. The LSP algorithm and the KPQ algorithm use the combination of principal curvature and Gaussian curvature at the key points to form the shape index value Si and the key point quality $Q$ as the rating indicators. Like the ISS algorithm, the neighborhood distribution characteristics of features are not considered when selecting key points, and there is no smoothing process for noise, so the repetition rate is lower when the noise intensity is high.

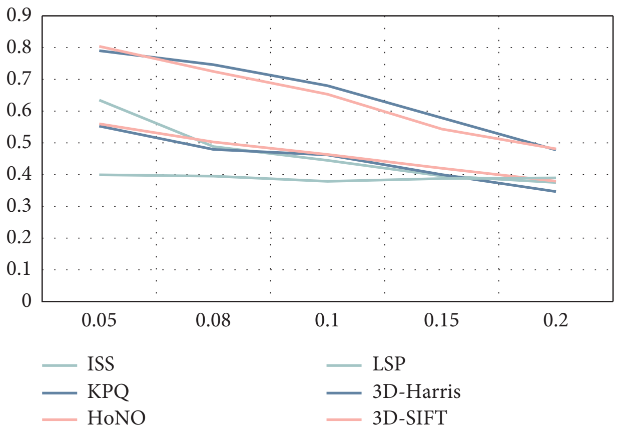

Figure 5: Gaussian noise-repetition rate change curve.

Table 6: The resolution repetition rate $r_{\text{density}}$ of the key point detection algorithm on the laser scanner data set.

| Reduced proportions | 50% | 70% | 80% | 90% | 95% |
|---|---|---|---|---|---|
| ISS | 0.096455 | 0.108979 | 0.094031 | 0.090092 | 0.066357 |
| KPQ | 0.028785 | 0.019695 | 0.017069 | 0.013635 | 0.013736 |
| HoNO | 0.152611 | 0.100899 | 0.07474 | 0.039087 | 0.027775 |
| LSP | 0.247046 | 0.188466 | 0.149884 | 0.087264 | 0.043834 |
| 3D-Harris | 0.084941 | 0.087567 | 0.088375 | 0.059388 | 0.046258 |
| 3D-SIFT | 0.1212 | 0.12827 | 0.102717 | 0.069286 | 0.047975 |

### 5.3. Resolution Repetition Rate $r_{density}$.

The original point cloud is reduced according to the reduction rate of 50%, 70%, 80%, 90%, and 95%. The key points before and after the reduction are detected, and the repetition rate is calculated. The $r_{\text{density}}$ on the 4 data sets are shown in Tables 6–9, respectively. The reduction rate-repetition rate curve shown in Figure 6 is drawn based on the average value of the $r_{\text{density}}$ on the 4 data sets.

From the data in the table, it can be found that increasing the neighborhood radius can make the algorithm more robust to resolution changes.

### 5.4. Outlier Repetition Rate $r_{outlier}$.

The original point cloud adds outliers according to the ratio of 0.1%, 0.5%, 1%, 2%, and 5%. The key points before and after the outlier are added, and the repetition rate is calculated. The $r_{\text{outlier}}$ on the 4 data sets are shown in Tables 10–13, respectively. The reduction rate-repetition rate curve shown in Figure 7 is drawn based on the $r_{\text{outlier}}$ average value on the 4 data sets.

In the above experiment, the key point detection algorithm decreases more as the proportion of outliers increases,

Table 7: The resolution repetition rate $r_{\text{density}}$ of the key point detection algorithm on the random view data set.

| Reduced proportions | 50% | 70% | 80% | 90% | 95% |
|---|---|---|---|---|---|
| ISS | 0.148672 | 0.161701 | 0.145844 | 0.085244 | 0.033532 |
| KPQ | 0.051409 | 0.030603 | 0.018382 | 0.029997 | 0.030603 |
| HoNO | 0.066155 | 0.046561 | 0.03232 | 0.019493 | 0.015857 |
| LSP | 0.086759 | 0.062822 | 0.036764 | 0.015655 | 0.003838 |
| 3D-Harris | 0.089991 | 0.081406 | 0.104434 | 0.055247 | 0.016059 |
| 3D-SIFT | 0.111908 | 0.122917 | 0.110999 | 0.082921 | 0.037269 |

Table 8: The resolution repetition rate $r_{\text{density}}$ of the key point detection algorithm on the Space Time data set.

| Reduced proportions | 50% | 70% | 80% | 90% | 95% |
|---|---|---|---|---|---|
| ISS | 0.117362 | 0.097061 | 0.079184 | 0.058075 | 0.040905 |
| KPQ | 0.033835 | 0.025149 | 0.012524 | 0.010403 | 0.007272 |
| HoNO | 0.134229 | 0.101303 | 0.068882 | 0.039693 | 0.022725 |
| LSP | 0.156348 | 0.114231 | 0.096354 | 0.044238 | 0.022422 |
| 3D-Harris | 0.094132 | 0.082012 | 0.081406 | 0.067973 | 0.035754 |
| 3D-SIFT | 0.145743 | 0.134128 | 0.104636 | 0.087163 | 0.043733 |

Table 9: The resolution repetition rate $r_{\text{density}}$ of the key point detection algorithm on the Kinect data set.

| Reduced proportions | 50% | 70% | 80% | 90% | 95% |
|---|---|---|---|---|---|
| ISS | 0.151096 | 0.142511 | 0.12726 | 0.089385 | 0.05858 |
| KPQ | 0.060398 | 0.061812 | 0.050601 | 0.049288 | 0.049086 |
| HoNO | 0.119382 | 0.077467 | 0.058378 | 0.03333 | 0.021816 |
| LSP | 0.205636 | 0.164327 | 0.123826 | 0.071407 | 0.038683 |
| 3D-Harris | 0.109282 | 0.10908 | 0.099384 | 0.063731 | 0.033229 |
| 3D-SIFT | 0.15958 | 0.15049 | 0.123523 | 0.091809 | 0.057873 |

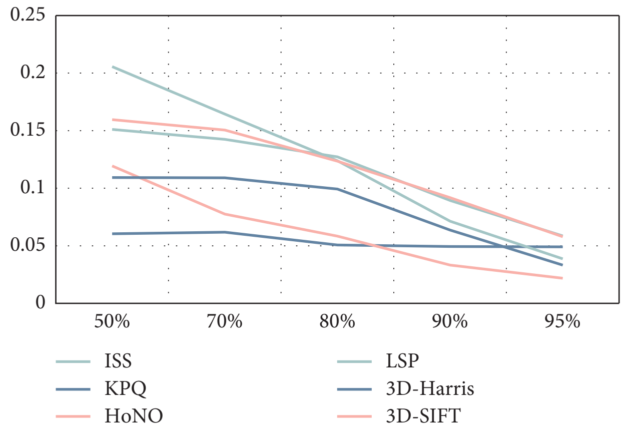

Figure 6: The relationship between the repetition rate and the reduction rate.

Table 10: The outlier repetition rate $r_{\text{outlier}}$ of the key point detection algorithm on the laser scanner data set.

| Proportion of outliers | 0.1% | 0.5% | 1% | 2% | 5% |
|---|---|---|---|---|---|
| ISS | 0.984851 | 0.903647 | 0.8787 | 0.841431 | 0.767196 |
| KPQ | 0.969398 | 0.846683 | 0.77063 | 0.734371 | 0.629836 |
| HoNO | 0.970812 | 0.913747 | 0.874256 | 0.833755 | 0.798001 |
| LSP | 0.994749 | 0.943138 | 0.887386 | 0.821837 | 0.705081 |
| 3D-Harris | 0.999999 | 0.933745 | 0.923039 | 0.883245 | 0.828604 |
| 3D-SIFT | 0.994749 | 0.948996 | 0.904758 | 0.865267 | 0.828604 |

Table 11: The outlier repetition rate $r_{\text{outlier}}$ of the key point detection algorithm on the random view data set.

| Proportion of outliers | 0.1% | 0.5% | 1% | 2% | 5% |
|---|---|---|---|---|---|
| ISS | 0.997678 | 0.949804 | 0.910212 | 0.856682 | 0.764267 |
| KPQ | 0.98778 | 0.893951 | 0.849309 | 0.742451 | 0.611757 |
| HoNO | 0.934654 | 0.770226 | 0.63125 | 0.519342 | 0.406727 |
| LSP | 0.994143 | 0.924251 | 0.872337 | 0.76053 | 0.557318 |
| 3D-Harris | 0.999999 | 0.976266 | 0.93021 | 0.8787 | 0.783962 |
| 3D-SIFT | 0.995759 | 0.953541 | 0.940512 | 0.90799 | 0.842037 |

Table 12: The outlier repetition rate $r_{\text{outlier}}$ of the key point detection algorithm on the space time data set.

| Proportion of outliers | 0.1% | 0.5% | 1% | 2% | 5% |
|---|---|---|---|---|---|
| ISS | 0.999999 | 0.97263 | 0.952733 | 0.9191 | 0.86153 |
| KPQ | 0.984952 | 0.939805 | 0.878902 | 0.827796 | 0.735078 |
| HoNO | 0.984851 | 0.896476 | 0.847289 | 0.802243 | 0.739118 |
| LSP | 0.998082 | 0.958389 | 0.904657 | 0.834866 | 0.681548 |
| 3D-Harris | 0.999999 | 0.996163 | 0.986669 | 0.961015 | 0.926574 |
| 3D-SIFT | 0.999999 | 0.971115 | 0.952026 | 0.937583 | 0.911626 |

Table 13: The outlier repetition rate $r_{\text{outlier}}$ of the key point detection algorithm on the Kinect data set.

| Proportion of outliers | 0.1% | 0.5% | 1% | 2% | 5% |
|---|---|---|---|---|---|
| ISS | 0.999999 | 0.977478 | 0.958793 | 0.939906 | 0.886982 |
| KPQ | 0.999496 | 0.962429 | 0.921423 | 0.865974 | 0.771943 |
| HoNO | 0.983134 | 0.901122 | 0.844764 | 0.776488 | 0.699526 |
| LSP | 0.999799 | 0.960914 | 0.922837 | 0.855672 | 0.718413 |
| 3D-Harris | 0.999999 | 0.99788 | 0.989396 | 0.965156 | 0.933038 |
| 3D-SIFT | 0.999999 | 0.981215 | 0.962833 | 0.942532 | 0.910818 |

indicating that the method of covariance matrix eigenvalue decomposition has stronger stability to outliers.

The above research verifies that the sports video moving target detection and tracking based on the SIFT algorithm proposed in this paper has good results.

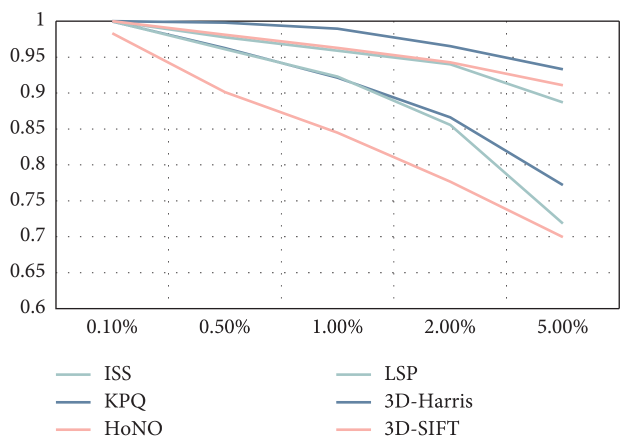

FIGURE 7: The relationship between the repetition rate and the proportion of outliers.

## 6. Conclusion

In the sports behavior recognition system, many algorithms lack timing information. In order to make up for the deficiencies of these algorithms, many researchers make up for the lack of time information by establishing a time series model for feature descriptors to further describe the behaviors and reflect the information that different types of behaviors change in chronological order so that the expression of features is more accurate and distinguishable, and the classification accuracy is improved. This paper uses SIFT to perform the recognition and analysis of sports video images and uses SIFT algorithm to recognize and track moving targets. Finally, this paper verifies through experiments that the sports video moving target detection and tracking proposed in this paper based on the SIFT algorithm has good results.

## Data Availability

The labeled dataset used to support the findings of this study is available from the corresponding author upon request.

## Conflicts of Interest

The authors declare no competing interests.

## Acknowledgments

This study was sponsored by Shenyang Jianzhu University.

## References

[1] M. Li, Z. Zhou, and X. Liu, “Multi-person pose estimation using bounding box constraint and LSTM,” *IEEE Transactions on Multimedia*, vol. 21, no. 10, pp. 2653–2663, 2019.

[2] J. Xu, K. Tasaka, and M. Yamaguchi, “Fast and accurate whole-body pose estimation in the wild and its applications,” *ITE Transactions on Media Technology and Applications*, vol. 9, no. 1, pp. 63–70, 2021.

[3] G. Szűcs and B. Tamás, “Body part extraction and pose estimation method in rowing videos,” *Journal of Computing and Information Technology*, vol. 26, no. 1, pp. 29–43, 2018.

[4] R. Gu, G. Wang, Z. Jiang, and J. N. Hwang, “Multi-person hierarchical 3d pose estimation in natural videos,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 30, no. 11, pp. 4245–4257, 2019.

[5] M. Nasr, H. Ayman, N. Ebrahim, R. Osama, N. Mosaad, and A. Mounir, “Realtime multi-person 2D pose estimation,” *International Journal of Advanced Networking and Applications*, vol. 11, no. 6, pp. 4501–4508, 2020.

[6] N. T. Thành and P. T. Công, “An evaluation of pose estimation in video of traditional martial arts presentation,” *Journal of Research and Development on Information and Communication Technology*, vol. 2, pp. 114–126, 2019.

[7] I. Petrov, V. Shakhuro, and A. Konushin, “Deep probabilistic human pose estimation,” *IET Computer Vision*, vol. 12, no. 5, pp. 578–585, 2018.

[8] G. Hua, L. Li, and S. Liu, “Multipath affinage stacked—hourglass networks for human pose estimation,” *Frontiers of Computer Science*, vol. 14, no. 4, pp. 1–12, 2020.

[9] K. Aso, D. H. Hwang, and H. Koike, “Portable 3D human pose estimation for human-human interaction using a chest-mounted fisheye camera,” in *Proceedings of the Augmented Humans Conference AHs 2021*, pp. 116–120, Rovaniemi, Finland, February 2021.

[10] D. Mehta, S. Sridhar, O. Sotnychenko et al., “Vnect: real-time 3d human pose estimation with a single RGB camera,” *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 1–14, 2017.

[11] S. Liu, Y. Li, and G. Hua, “Human pose estimation in video via structured space learning and halfway temporal evaluation,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 29, no. 7, pp. 2029–2038, 2018.

[12] S. Ershadi-Nasab, E. Noury, S. Kasaei, and E. Sanaei, “Multiple human 3d pose estimation from multiview images,” *Multimedia Tools and Applications*, vol. 77, no. 12, pp. 15573–15601, 2018.

[13] X. Nie, J. Feng, J. Xing, S. Xiao, and S. Yan, “Hierarchical contextual refinement networks for human pose estimation,” *IEEE Transactions on Image Processing*, vol. 28, no. 2, pp. 924–936, 2018.

[14] Y. Nie, J. Lee, S. Yoon, and D. S. Park, “A multi-stage convolution machine with scaling and dilation for human pose estimation,” *KSII Transactions on Internet and Information Systems (TIIS)*, vol. 13, no. 6, pp. 3182–3198, 2019.

[15] A. Zarkeshev and C. Csiszár, “Rescue method based on V2X communication and human pose estimation,” *Periodica Polytechnica: Civil Engineering*, vol. 63, no. 4, pp. 1139–1146, 2019.

[16] W. McNally, A. Wong, and J. McPhee, “Action recognition using deep convolutional neural networks and compressed spatio-temporal pose encodings,” *Journal of Computational Vision and Imaging Systems*, vol. 4, no. 1, p. 3, 2018.

[17] R. G. Díaz, F. Laamarti, and A. El Saddik, “DTCoach: your digital twin coach on the edge during COVID-19 and beyond,” *IEEE Instrumentation and Measurement Magazine*, vol. 24, no. 6, pp. 22–28, 2021.

[18] A. Bakshi, D. Sheikh, Y. Ansari, C. Sharma, and H. Naik, “Pose estimate based yoga instructor,” *International Journal of Recent Advances in Multidisciplinary Topics*, vol. 2, no. 2, pp. 70–73, 2021.

[19] S. L. Colyer, M. Evans, D. P. Cosker, and A. I. Salo, “A review of the evolution of vision-based motion analysis and the integration of advanced computer vision methods towards developing a markerless system,” *Sports Medicine-open*, vol. 4, no. 1, pp. 1–15, 2018.

[20] I. Sárándi, T. Linder, K. O. Arras, and B. Leibe, “Metrabs: metric-scale truncation-robust heatmaps for absolute 3d human pose estimation,” *IEEE Transactions on Biometrics, Behavior, and Identity Science*, vol. 3, no. 1, pp. 16–30, 2020.

[21] A. Azhand, S. Rabe, S. Müller, I. Sattler, and A. Heimann-Steinert, “Algorithm based on one monocular video delivers highly valid and reliable gait parameters,” *Scientific Reports*, vol. 11, no. 1, pp. 1–10, 2021.

[22] J. Xu and K. Tasaka, “Keep your eye on the ball: detection of kicking motions in multi-view 4K soccer videos,” *ITE Transactions on Media Technology and Applications*, vol. 8, no. 2, pp. 81–88, 2020.