



Action recognition by key trajectories

Fernando Camarena¹ · Leonardo Chang¹ · Miguel Gonzalez-Mendoza¹ · Ricardo J Cuevas-Ascencio¹

Received: 22 September 2020 / Accepted: 13 December 2021 / Published online: 20 January 2022 / Published online: 20 January 2022
© The Author(s) 2022, corrected publication 2022

Abstract

Human action recognition is an active field of research that intends to explain what a subject is doing in an input video. Deep learning architectures serve as the foundation for cutting-edge approaches. Recent research, on the other hand, indicates that hand-crafted characteristics are complementary and, when combined, can enhance classification accuracy. Cutting-edge approaches are based on deep learning architectures. Recent research, however, indicates that hand-crafted features complement each other and can help boost classification accuracy when combined. We introduce the key trajectories approach that is based on the popular, hand-crafted method, improved dense trajectories. Our work explores how pose estimation can be used to find meaningful key points to reduce computational time, undesired noise, and to guarantee a stable frame processing rate. Furthermore, we tested how feature-tracking behaves with dense inverse search and with a frame to frame subject key point estimation. Our proposal was tested on the KTH and UCF11 datasets employing Bag-of-words and on the UCF50 and HMDB datasets using Fisher Vector, where we got an accuracy performance of 95.71, 84.88, 92.9, and 81.3%, respectively. Also, our proposal can recognize subject actions in video eight times faster compared to its dense counterpart. To maximize the bag-of-words classification performance, we illustrate how the hyperparameters affect both accuracy and computation time. Precisely, we present an exploration of the vocabulary size, the SVM hyperparameter, the descriptor's distinctiveness, and the subject body key points.

Keywords Action recognition · Pose estimation · Dense trajectories · Key trajectories

1 Introduction

Human action recognition is a fundamental task of intelligent video-surveillance systems that seek to understand what the subjects are doing in an input video.

This task can be addressed by two main approaches: hand-crafted methods and deep learning architectures. Although deep learning for action recognition has reported high accuracy performance [15, 52] in common datasets [28, 35,

41, 48], recent work [8, 23, 48] suggests that hand-crafted features are complementary to deep learning methods, and together can improve the classification performance.

Our findings show that improved dense trajectories achieve high classification performance and can be enhanced using deep learning procedures. Common hand-crafted methods, like improved dense trajectories, are divided into two main tasks: Action Representation and Action Classification. Action Representation involves 3 steps: sampling, feature extraction, and feature encoding.

Sampling aims to locate the frame regions to be analyzed, it can be performed both densely or by finding interest points. Then, the feature extraction step takes these points to get relevant information about the video. Finally, feature encoding discretizes the entire space of local features extracted from a training set, common methods comprise bag-of-words (BoW) [7] and fisher vector (FV) [32].

On the other hand, action classification concerns how to distinguish one action from another, according to the set of features collected in previous steps. Action classification can be divided into three main approaches: template-based,

✉ Fernando Camarena
fernando@camarenat.com

Leonardo Chang
lchang@tec.mx

Miguel Gonzalez-Mendoza
mgonza@tec.mx

Ricardo J Cuevas-Ascencio
rjcuevasa@gmail.com

¹ School of Engineering and Science, Tecnológico de Monterrey, Campus Estado de México, Atizapán de Zaragoza, Estado de México, Mexico

generative, and discriminative models. A detailed explanation can be found in Sect. 2.

This paper introduces the *Key Trajectories approach* that describes an action using subject key points estimation and dense trajectories. We tested our proposal on four publicly available datasets for subject action recognition (KTH, UCF11, UCF50, and HMDB51) and found that subject key points are a viable alternative for the detection of interest points in the sampling step and can be a faster replacement for human optical flow.

Using an Nvidia 1080TI GPU for pose estimation, Key Trajectories accomplishes the action recognition task 8 times faster, generating about 80–90% fewer trajectories, with little to none impact on the recognition performance, i.e., 94.20% compared to the 95.65% of the dense trajectories in the KTH dataset. Regarding the UCF11 dataset, we got an accuracy of 80.66% in contrast to the 84.08% of its dense counterpart. We propose a second configuration that recognizes actions two times faster, but accuracy is not affected. The results achieved were 95.71 and 84.43% for the KTH and UCF11, respectively.

To better understand our results, we examined how hyperparameters affect both accuracy and computation time. Specifically, we provided an exploration of the bag-of-words vocabulary size, the SVM hyperparameter, the amount of the information supplied by the descriptors, and the subject body key points. At the tracking step, we explored how the use of dense inverse search affects computation time and how to track using pose estimation.

Finally, we tested our method using Fisher Vector in the UCF50 and HMDB51 datasets, where we achieved 92.9 and 81.3%, respectively, compared to 91.2 and 57.2% of the dense trajectory procedure.

We organized the rest of this paper as follows: Section 2 provides an overview of pertinent methods for the action recognition task. Then, in Sect. 3, we identify which pieces of the process needlessly increase the computation time, and we introduce our proposal. In Sect. 4, we report the experiments together our results in in Sect. 5 (Hyperparameters analysis), Sect. 6 (Efficiency analysis) and Sect. 7 (Effectiveness analysis). Finally, in Sect. 8, we establish the conclusions of this research.

2 Related work

Human Activity Recognition (HAR) [52] can be categorized into 3 levels: gestures, actions, and interactions; gestures are atomic body movements, actions are sequences of gestures with a linked meaning, and interactions are actions comprising two or more doers. Our work focuses on action

recognition, which can be divided into two subtasks: action representation and action classification.

2.1 Action representation

Action representation has been approached in three distinct ways: global, local, and depth-based representations [52]. Global representations are not robust to occlusions, noise, and changing viewpoints, so they have fallen into disuse over time. Depth-based representations need knowledge about the object depth, which can be obtained by employing special cameras like Microsoft Kinect.

On the other hand, local representations [52] aim to describe a video through a combination of local descriptors sampled densely (by placing points without any semantics) or by locating points of interest. There are several approaches for the location of points of interest [3, 14, 22], but one of the most widely used is space-time interest points (STIP) [22], which is an extension of the Harris detector [14]. Although detecting interesting points may improve performance, there are some issues to consider, such as the number of points to be used and their representativeness [52].

Once the interest points have been identified, the next step is to describe them. The most widely used methods are Scale-invariant feature transform (SIFT) [25, 26], 3D SIFT [36], speed-up robust features (SURF) [1], 3D SURF [50], HOG [9], HOF [10], and MBH [44].

It is possible to combine several descriptors to improve performance, such as the dense trajectory approach [44, 45] that achieves high accuracy through the descriptors HOG, HOF, MBH, and the displacements of the trajectories (DT). Shi et al. [37] present a sequential deep trajectory descriptor (sDTD) to capture long-term motion information.

The next step is feature encoding, which the key idea is to discretize the entire space of local features extracted from a training set. Several works have been presented: bag-of-words (BoW) [7], fisher vector (FV) [32], stacked fisher vector (SFV) [31], vector quantization (VQ) [40], vector of locally aggregated descriptors (VLAD) [17], super vector encoding (SVC) [53]. According to [52], the best performance is achieved by using dense trajectories with SFV.

2.2 Action classification

Action classification [52] has been carried out using 3 approaches: template-based, generative, and discriminative models. The simplest models are template-based, with the principle of finding the closest of a set of predefined templates. Generative methods use probabilistic procedures like the Hidden Markov Model (HMM) [4], Dynamic Bayesian networks (DBN) [12]. Discriminative ones use machine learning techniques like support vector machines (SVM)

[30], conditional random fields (CRF) [42], and deep learning architectures [52].

On the other hand, deep learning architectures [52] like deep neural networks (DNNs) [2, 16], convolutional neural networks (CNNs) [23, 29, 39], and recurrent neural networks (RNNs) [13, 43] have achieved high accuracy values and even many of them perform the action recognition task in real-time [27, 51]. Unlike traditional machine learning methods, deep neural networks can learn representations automatically.

State-of-the-art methods focus on improving classification performance by combining CNN features with hand-crafted features. Li et al. [23] proposed to combine CNN with VLAD to capture mid-range and long-range dynamics. Wang et al. [48] presented the deep-convolutional descriptor which combines dense trajectories with CNN features. Chéron et al. [8] introduced a Pose-based Convolutional Neural Network descriptor that proved that CNN approaches are complementary to hand-crafted approaches.

Jhuang et al. [18] examine the limits of recognition algorithms using ground truth information to identify what factors most influence the model's performance. This work [18] found that mid-to-high level features had the greatest influence on classification accuracy. Based on the Jhuang et al. [18] work and the current success of deep learning methods, we suggest a new way to combine CNN approaches with hand-crafted procedures by using CNN to locate meaningful key points to reduce computational time, undesired noise, and guarantee a stable frame processing rate, which can be applied to challenging datasets.

3 Our proposal: action recognition by key trajectories

Algorithms must take into account both accuracy and computation time to allow real-time subject action recognition and its application in video-surveillance systems. Nevertheless, the trend is to build increasingly effective methods. So, our aim is to speed up the dense trajectories approach with no impact on recognition accuracy. In this section, we break down the action recognition task into different stages and sub-stages to explain each one of our contributions.

3.1 Key trajectories extraction

To extract trajectories from a video, dense trajectories broke down the recognition task into three consecutive steps: dense sampling, tracking, and description. This Section discusses pertinent factors that might result in needless computations that increase execution time.

3.1.1 Sampling

Dense trajectories [44, 46] suggest that sampling should be accomplished densely to create robust models to occlusions, scale and rotation variations, and other well-know problems in computer vision systems. Nevertheless, this idea leads to extract points on the whole image without any semantics, which may result in a bottleneck because feature-tracking requires the application of an optical flow algorithm to each point that can be highly expensive.

Furthermore, a trajectory is a motion description, but in a real-life video, there are several motion sources in addition to the object of interest, such as secondary objects, basic camera movements, environmental conditions, and so on. Hence, dense sampling implies considering all motion sources that result in unnecessary calculations and noise in the final model.

Our proposal hypothesizes that the subject key points, shown in Fig. 1, contain comparable distinctiveness and representativeness as using a dense sampling over several scales.

Considering only subject key points implies a constant number of the frame-locations used, which enables a stable frame processing rate, a critical feature for high-resolution and lengthy videos. In contrast to our proposal, in dense trajectories, the frame-locations used are highly related to the video-resolution and scales used. Additionally, the estimation of subject key points gives us the necessary information to group trajectories by people, which can be helpful for interaction recognition.

The estimation of subject key points must fulfill two properties: repeatability and high discrimination. Repeatability is necessary to the discovery of the same point over several object views that gives robustness to the scale, rotation, viewpoint, and illumination variations. Then, high discrimination ensures that a point has a binary connection with a subject. These properties are satisfied by the work of Cao et al. [6, 38, 49].

Our proposal examined the distinctiveness of subject key points to find a setup that could further reduce computation time and determined two sets of key points: using all the subject key points, as described in Fig. 1, and a set of essential key points described in Sect. 5.2.1. Figure 2 shows what the use of key points looks like and its comparison with its dense counterpart.

3.1.2 Tracking

Feature tracking is the process of determining the location of a point in successive frames, which results in a temporal-sequence for each point created during the sampling step.

The key idea to reduce the computational time is to analyze each point faster. Hence, we recommend the use of the

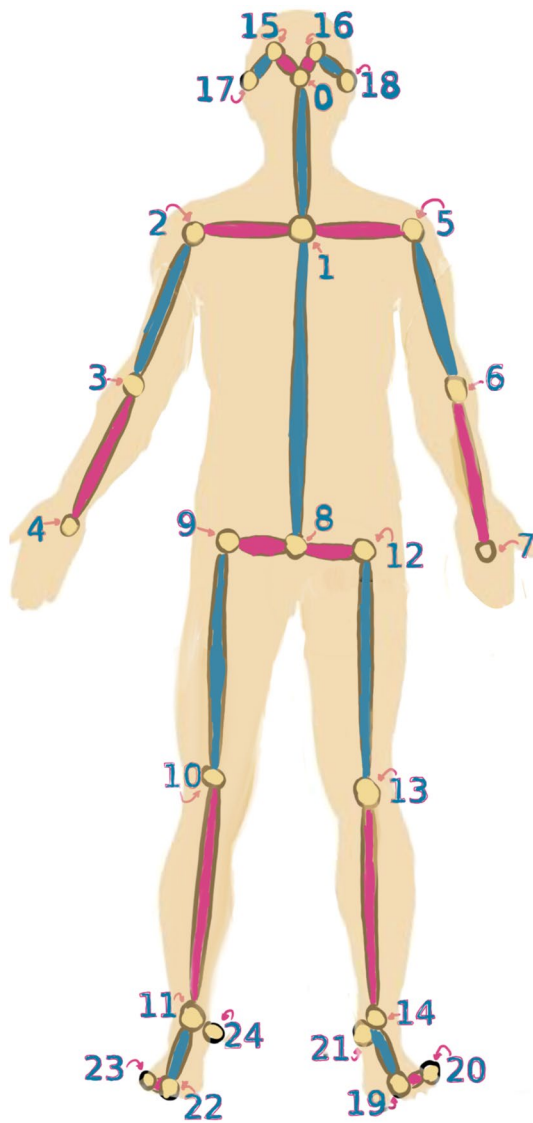


Fig. 1 Subject key points. A visual representation of the subject key points that we propose to use in our work [6, 38, 49]

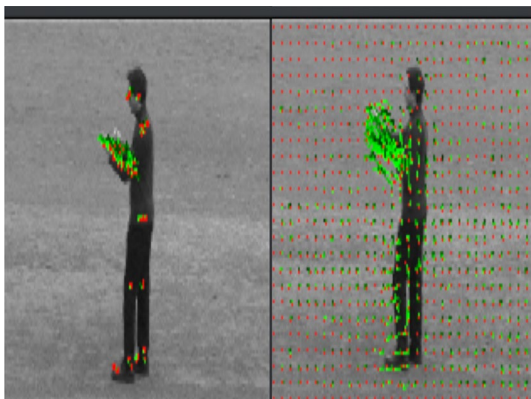


Fig. 2 subject key point sampling versus dense sampling. Using the subject's key points considerably decreases the amount of data to be analyzed

Kroeger et al. [19] optical flow algorithm, which outperforms in both accuracy and computation time to *Farneback* suggested by dense trajectories.

On the other hand, another advantage of the subject key points estimation is that it can be used to match points along each of the frames, leading to a faster form of feature-tracking. Nevertheless, some points need to be taken into account:

1. A full-body view of the subject is required in the video.
2. Occlusions may have an impact on the detection process.

To address the concerns raised above, we made the following decisions:

1. If a key point does not exist in the following frame (due to occlusion), the preceding frame's value is utilized instead.
2. If the full-body view is missing, we performed the optical flow algorithm instead.

Other parameters in our study, such as the trajectory length and frame step size, correspond to those in the Wang et al. [45] paper, which demonstrated greater accuracy and avoided drifting problems.

3.1.3 Description

The description of the sequences of points extracted from a video is the last step needed to complete the extraction process. Wang et al. [45] proposed to use HOG [9], HOF [10], MBH (MBHX, MBHY) [45], and trajectory displacements (DT) [44].

Using a large array of descriptors can produce valuable knowledge about video-motion. However, each descriptor appends 4000 features to the final video descriptor. Hence, dense trajectories generate a 20,000-dimensional vector, which implies dealing with high dimensionality-related issues.

The estimation of subject key points means working with a lower number of trajectories. As an example, dense trajectories extract about 527 trajectories from a 160×120 -video of 131 frames, despite key trajectories that only obtain 35 trajectories, which is just 10% of the number of its dense counterpart. So, using a lower number of features per descriptor seems feasible. Furthermore, an assessment of the distinctiveness of each descriptor (TRAJ, HOG, HOF, MBHX, MBHY) is presented in Sect. 3.2 to discover how descriptors affect classification accuracy.

3.2 Key trajectory, feature encoding

Feature encoding is accomplished with bag-of-words (BoW) using *k*-means, but other clustering algorithms

could be used as well. [34]. Reducing the computational time of the vocabulary constructions has a two-fold contribution: hyper-parameter optimization and faster descriptor creation.

First, being able to build vocabularies faster enables us to carry out the hyper-parameter analysis presented in this paper using our computer system. Second, the final descriptors are created by associating each trajectory by comparison to a corresponding visual word of the vocabulary. Hence, the vocabulary size (the number of visual words that constitute the vocabulary), is correlated to the number of comparisons made, which affects both the accuracy and the computation time of the classification stage.

As mentioned before, key trajectories work with a lower number of trajectories, which implies dealing with sparse vectors if we use the 4000-size as Wang et al. proposed. Therefore, it is not unreasonable to think that a smaller vocabulary is more appropriate than a larger one.

To answer our hypothesis, we explored different vocabulary sizes to determine which fit better with our aim. Also, we introduce a novel way, shown in Fig. 3, to associate trajectories and construct a descriptor using a reduced number of subject key points. The general idea is that the injective relationship of a trajectory to a subject key point can be used to group trajectories. The first step is to create a 250-size vocabulary that means a reduction of 96% compared to the dense trajectories. Next, a descriptor for each group is formed, and its concatenation is the final descriptor. Our results show that this scheme helps to maximize accuracy with no impact on the computation time.

4 Experimental results

This Section aims to introduce the experiments conducted to assess our proposal. All experiments were carried out using a personal computer with macOS High Sierra, an Intel Core i7 processor (seventh generation at 2.9GHz), and 16 GB of RAM except for the subject key points estimation that was obtained using an Nvidia GTX 1080TI graphics card.

These trajectories [45] were obtained using the C++ implementation of the method provided by the authors using the protocol described in their paper [46]. For key trajectories, all algorithms were developed in C++ using OpenCV 3.4.1.

The vocabulary constructions were implemented using the work of Chang et al. [7] with the configuration described by Wang et al. [45] (4000 visual words, one vocabulary per descriptor).

The kernel used for the SVM was the χ^2 kernel, available in OpenCV, but one-class classifiers could be also used for abnormal behavior recognition [5]. Then, for the dense inverse search algorithm, we use the implementation available in the OpenCV library.

Finally, the subject key points estimation was performed using the work and setup proposed by Cao et al. [6, 38, 49].

4.1 Datasets

We use accuracy to assess the effectiveness of our experiments conducted on the KTH [35] (6 action classes), UCF11 [24] (11 action classes), UCF50 [33] (50 action classes), and HMDB51 [20] (51 action classes) datasets. using the

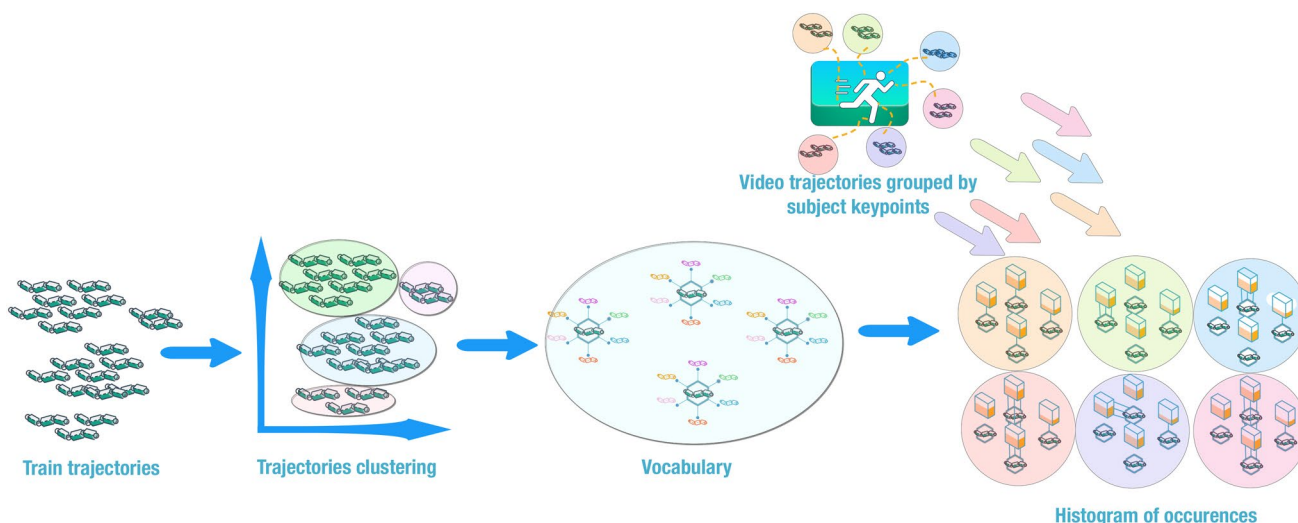


Fig. 3 A new way to encode trajectories. This Figure shows our new way to encode a trajectory using a small vocabulary. The subject key points are used to form a total of 6 groups (or the number of subject

key points contemplated) and employing a general vocabulary get the six mini descriptors, whose concatenation will form the final video descriptor

original protocol provided by the authors. On the other hand, experiments related to reducing the computational time were tested in the KTH dataset.

4.2 Experiments setup

As previously mentioned, we identified that both classification accuracy and computational time are sensitive to specific parameters. So, to explore the solution space and find a trade-off between these, we conducted two sets of experiments.

The first one is intended to study the hyperparameters involved in action recognition. Concretely, this experiment comprise:

Bag-of-words size. The number of visual words that form a vocabulary impacts both accuracy and computation time. On one hand, a large number of visual words imply better results, but the efficiency might be affected. So, the idea is to find a middle point that provides an adequate trade-off between efficiency and effectiveness. To measure the computation time related to the descriptor and vocabulary

constructions, we took the time in seconds normalized using the number of trajectories involved.

Descriptors distinctiveness. The dimensionality of a descriptor is correlated with the computation time and classification accuracy. A high dimensionality vector can improve results but can easily become a bottleneck. We investigated the C and gamma SVM parameters using *chi2* and *RBF* kernels to better understand each descriptor. To measure the SVM computation time, we took the average time that it takes to classify a descriptor in the testing set.

Subject key points. The number of subject key points used has a strong relationship with classification accuracy and computation time. So, we assess the distinctiveness of each point using classification accuracy to know which points represent better an action.

The second set of experiments is intended to assess the effectiveness of our proposal using accuracy as the evaluation metric. We compare three configurations of our proposal together with dense trajectories.

Each configuration is goal-specific: A and B for applications that expect high accuracy performance, and C if timing is a critical issue. Figure 4 summarizes each configuration.






	Wang et al.	Our proposal A (Disof + op25 + 5 Desc)	Our proposal B (Disof + op25+ 3 Desc)	Our proposal C (op6 + 3 Desc)
Sampling : 	$S^* [(w/st - 1) (h/st-1)]$ See Equation 2, example: Res: 240x160, st=5 and s=4 ->5,828 points	25 points	25 points	6 points
Tracking : 	Farneback	Disof	Disof	Keypoints estimation
Description : 	TRAJ + HOG + HOF + MBHX + MBHY	TRAJ + HOG + HOF + MBHX + MBHY	HOG + MBHX + MBHY	HOG + MBHX + MBHY
Encoding : 	5 vocabularies of 4,000 vectors	5 vocabularies of 1,000 vectors	3 vocabularies of 1,000 vectors	2 vocabularies of 250 vectors and 1 vocabulay of 200 vectors
Descriptor : 	20,000 features per video	5,000 features per video	3,000 features per video	4,200 features per video
Accuracy :	95,65%	94,9%	<u>95,71%</u>	94,2%
Speed-up:	-	1.78x	<u>1.79x</u>	<u>7,9x</u>

Fig. 4 Key Trajectories. Our proposal is composed of three different configurations. Our proposal $A_{Disof + op_{25} + 5 desc}$ aims to reduce the number of features that compose the final vector by reducing the number of visual words. Our proposal $B_{Disof + op_{25} + 3 desc}$ has the

purpose of testing the best subset of descriptors found. Our proposal $C_{OP + op_{6} + 3 desc}$ tests our new idea to encode trajectories using very small vocabularies and the estimation of points to replace the optical flow process

Also, We also experimented with a Fisher Vector version of the configuration A.

Configuration A introduces the estimation of subject key points, the implementation of dense inverse search as the optical flow algorithm. This version of our proposal is tested using bag-of-words and fisher vector.

Configuration B has the same setup that configuration A, but it uses a lower array of descriptors (HOG, MBHX, MBHY) formed by smaller vocabulary.

Configuration C prioritizes computational time over classification accuracy. It takes the same setup that configuration B, but the set of six-subject crucial points, identified in the first experiments set, and implements our encoding ideas outlined in Sect. 3. Also, we include the frame-to-frame estimation of subject key points as a feature-tracking.

For each of these bow-configurations, together with the Wang et al. [45] method, we assess:

1. Classification Accuracy
2. Speed-up for each step of the action recognition process.
3. Average number of trajectories extracted.
4. Computational time for vocabulary construction.
5. Computational time for the SVM model.
6. Overall computational time (trajectories extraction, descriptor construction, classification task), shown in the Eq. (3).

For FV-configurations, we only provided classification accuracy.

Equation 2 shows the number of frame locations analyzed when sampling is done densely. In contrast, our proposal uses a fixed number of subject key points.

$$FT = \left(\sum_{i=1}^n \frac{t_i}{f_i} \right) / n \tag{1}$$

Where FT is the normalized time to process a frame, n is the number of instances, t_i is the required time to process the instance i , f_i is the number of frames that has the instance i .

$$P = S * (wst - 1) * (hst - 1) \tag{2}$$

Where S is the number of scales used. w and h are the frame width and height, respectively, st is the sampling step size, and P is the number of frame locations.

For feature tracking, acceleration is measured using the time to process one frame, outlined in Eq. (1).

$$RT = TE + TD + TC \tag{3}$$

Where RT is the full time that is required to perform the recognition task, TE is the time needed to extract the trajectories of a video, TD is the time to build a descriptor from a previously generated vocabulary, and TC is the time needed to perform the classification process.

5 Hyperparameter analysis

This Section will examine the hyperparameter space in order to determine the optimal trade-off between accuracy and speed. We begin by reviewing the size of the BoW vocabulary, then investigate the SVM hyperparameters, and conclude with a distinctiveness analysis of the descriptors.

5.1 Bag-of-words analysis

Trying to determine the exact number of visual words that maximize both the distinctiveness and computation time of a descriptor is an open question. On one hand, using a big number of visual words improves classification accuracy, but also increases computational time. Furthermore, there exists a high risk of creating a model that overfits the training data. On the other hand, a smaller number of visual words benefit computational time, but the final model can underfit the training data.

Figure 5 shows how the size of the vocabulary impacts the classification accuracy for each video-descriptor. On the other hand, the computing time required to construct a descriptor is proportional to the number of processed trajectories. To get a value reflecting the influence of the vocabulary size, we normalize the time in seconds required by the number of trajectories; this allows us to determine if there is a link between the vocabulary size and the computing time, as seen in Fig. 6. The following are important points to consider about Figs. 5 and 6:

- Increasing the number of visual words used does not mean increasing the classification accuracy. Nevertheless, the computational cost always grows.
- A deficient number of visual words can generate underfitting models.

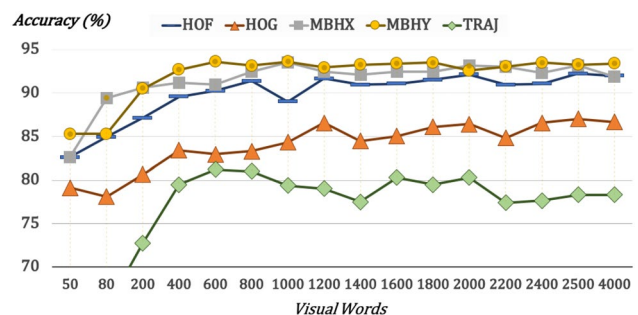


Fig. 5 Analysis of descriptors distinctiveness. Classification accuracy is not determined by the number of visual words to build vocabulary

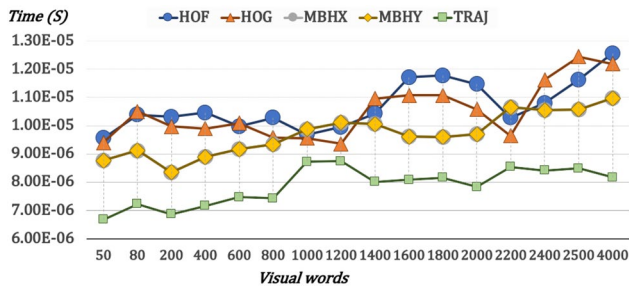


Fig. 6 The main idea of this figure is to answer if the vocabulary size is correlated with the time required to construct a descriptor, such as HOF or HOG. To get an accurate estimate of the impact of the vocabulary size, we normalized it by the number of trajectories. The findings show that increasing the vocabulary size has no noticeable effect on any of the descriptors. Hence, the only factor that impacts the execution time is the number of trajectories that were extracted from the video

- The accuracy performance is stabilized when reaching a value between 800-1000 visual words.
- MBHX, MBHY, and HOF performed the best in classification accuracy.
- TRAJ and HOG achieved the lowest performance
- the number of trajectories is a key factor that affects the computation time.

Another aspect where the vocabulary size impacts the computation time is in the construction of a descriptor. Figure 7 explores the relationship between the number of visual words and the SVM classifier and determines that the growth of both is proportional. Therefore, the more visual words we have, the longer the calculation time required.

Our results help us to understand better how action recognition is affected by its hyperparameters; dense trajectories [45] suggested to use 4000 visual words. Nevertheless, we

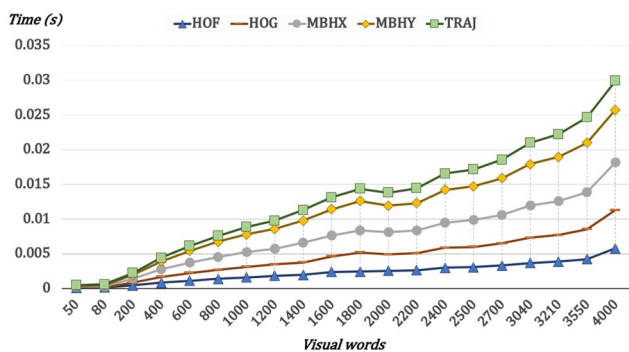


Fig. 7 BoW: vocabulary size versus classification time. This figure shows the computation time needed to perform the classification task using a descriptor. The most important conclusion is that the computation time grows proportionally when we increase the number of visual words used

found that in the KTH dataset, accuracy does not improve for values higher than 1000 visual words using key trajectories, but the computation time increased proportionally. Therefore, we set the number of visual words for key trajectories to 1000. This enabled us to achieve good performance in the KTH, UCF, and HMDB datasets.

5.2 Descriptor analysis

Action classification is carried out using an SVM due to its accuracy performance. Although dense trajectories reported good results, it's not clear what parameters they used and the reason for their choice. Hence, we explore the hyperparameter space (C and gamma).

Figure 8 explains our findings, where the color hue represents accuracy (lighter better), and each matrix position is a distinct combination of C and gamma parameters. The best results were achieved using a value of 0.01 for gamma and a range between 100 and 100,000 for the C parameter.

Another factor that is correlated to the computation time for the action classification is the vector dimensionality that represents a video. Dense trajectories proposed HOG, HOF, MBHX, MBHY, and TRAJ, which form a 20,000-dimensional descriptor (4000 by descriptor). Our findings suggested that 5000 features (1000 per descriptor) are enough to represent an action. But, now, we want to know if all the descriptors are essential or if we can omit one or more to reduce computation time with little to none impact on

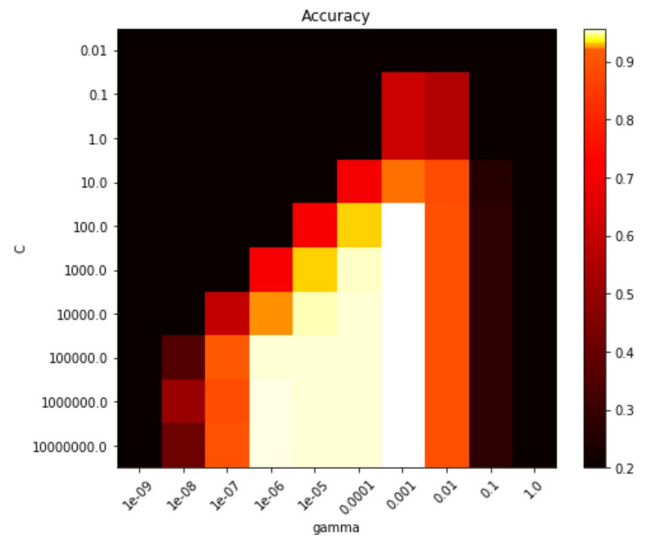


Fig. 8 SVM: hyper parameters. Each space of the matrix draws a different configuration of C and gamma, and the brightness of the color determines the accuracy (clearer, better). The best results are when we set gamma to a value of 0.01, and the value of C is in the range of 100–100,000.0. The SVM kernel is χ^2 using the HOG, MBHX, and MBHY descriptors

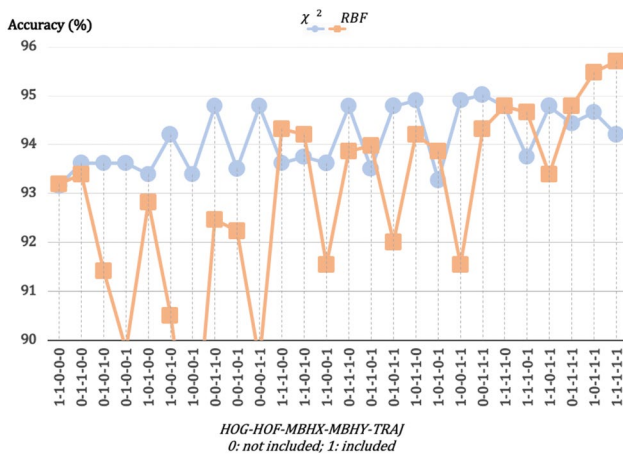


Fig. 9 SVM: descriptors Analysis (two or more descriptors). The blue line is the values when using a χ^2 kernel and the orange line using the *RBF* kernel. χ^2 kernel is more stable than the *RBF* kernel. Also, HOG, MBHX, and MBHY is the best descriptor subset

classification performance. So, we assess every combination of 2 or more descriptors using the χ^2 and *RBF* Kernels.

Figure 9 helps us to understand the descriptors and give us the next conclusions:

- χ^2 kernel showed stable performance in all the tests performed.
- *RBF* kernel achieves the best result, but it hasn't shown a stable performance.
- χ^2 kernel got its best results using only HOG, MBHX, and MBHY descriptors.
- *RBF* kernel got its best results using all descriptors

Hence, we found two novel configurations: using a kernel χ^2 with the HOG, MBHX, and MBHY descriptors (3000 features, i.e., 15% of the vector-size that dense trajectories proposed) and a configuration that uses the *RBF* kernel with all the descriptors, i.e., 5000 features.

5.2.1 Analysis of subject key points distinctiveness

As described in Sect. 3, sampling is the procedure that determines which places are going to be examined in order to gather information about the video. Carrying out the computation in a dense manner may result in needless computation. Hence, in this Section, we study the distinctiveness of the subject key points to know the impact of each point and group of points to the final classification accuracy.

To begin, we examine each key point by performing the recognition task on the KTH dataset using only the trajectories collected by the target body point, Although

using only the KTH for our analysis may be biased, we found that the model performance is consistent with more complicated datasets, as shown in Sect. 7. Our findings are shown in Fig. 11, as expected, using a single point has an impact on the model's performance. However, the classification accuracy was greater than 60 percent in several cases, demonstrating the critical role of the human body in identifying human-actions.

Our second set of experiments, shown in Fig. 10, examines how well a model performs when two or more key points are used. Due to combinatorial problems, we took the points in Fig. 11 that got an accuracy higher than 60%. Relevant insights include the following:

- The key point associated with the head is irrelevant for the purpose of action classification..
- The torso is crucial for proper action understanding.
- Hands play a critical role in describing an activity.
- The collection of points formed by the hands and torso enables an effective recognition performance.
- By adding forearm and ankle information, classification accuracy is improved.
- A smaller number of subject key points outperform the whole set of 25.

Therefore, if computing time is a constraint, we recommend using the subset formed by the hands, forearms, ankles, and torso.

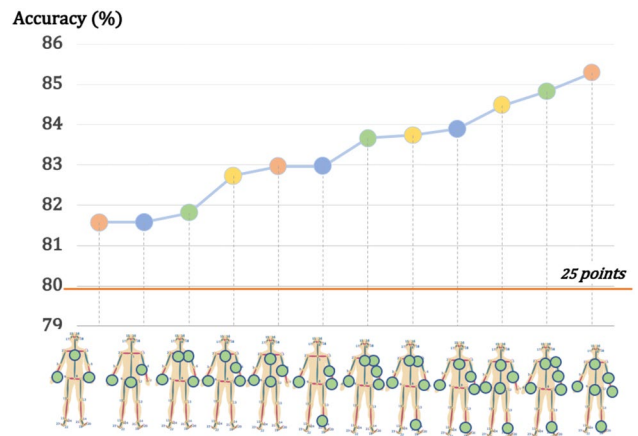


Fig. 10 Analysis of Subject key points distinctiveness. This Figure shows the best configurations when analyzing all possible combinations of 2 or more subject key points (greater than 60% by itself). It is possible to observe that the subject key points located in the arms and torso are those that have the best results and that adding the information of the legs increases the performance. The accuracy showed is using the TRAJ descriptor

Fig. 11 Subject key points analysis. This Figure shows the results obtained in the KTH dataset using a single point to find trajectories (shown on the right). In general, all points have similar accuracy. However, the hands generally excels in its performance

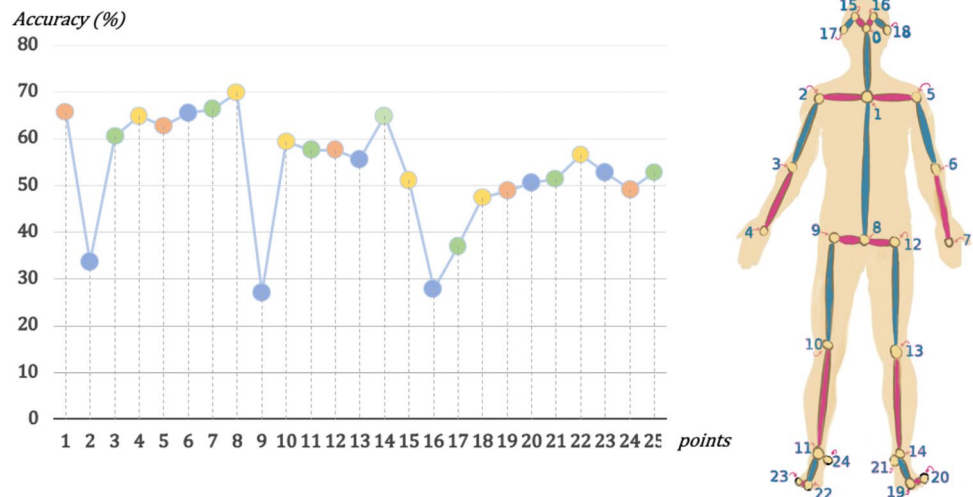


Table 1 Vocabulary construction: Computation time. The computational time required for each of the configurations presented together with the Wang et al. work is shown. Our proposal $C_{OP + op_6 + 3 desc}$ performs 13 times faster

	HOG (s)	HOF (s)	MBHX (s)	MBHY (s)	TRAJ (s)	TOTAL (s)
Wang et al. _{Farneback + 3 desc}	377.35	494.22	538.14	546.64	246.72	2203.10 (-)
Our proposal $A_{Disof + op_25 + 5 desc}$	169.26	166.57	176.87	233.43	162.17	908.32 (2.4x)
Our proposal $B_{Disof + op_25 + 3 desc}$	169.26	0	176.87	233.43	0	579.56 (3.8x)
Our proposal $C_{OP + op_6 + 3 desc}$	49.435	0	60.94	62.27	0	172.65 (12.8x)

6 Efficiency assessment

We aim to search through the hyperparameters space to find a trade-off between efficiency and effectiveness. In this Section, we review the efficiency of key trajectories compared to its dense counterpart [45]. The evaluation metrics are described in Sect. 3, and the dataset used is KTH.

First, this Section explains the behavior of our proposal in offline steps (Vocabulary construction and SVM model generation). Then, we explore the rest of the steps (trajectory extraction, descriptor construction, and classification).

6.1 Vocabulary construction

Earlier, we emphasized that the number of visual words used to construct the action vocabulary affects the computation time. So, we introduced a three-fold proposal, described in Sect. 4.2. Table 1 show the time in seconds required to perform the construction of the vocabulary for our proposals and dense trajectories. Some remarks are :

- Proposal $A_{Disof + op_25 + 5 desc}$ enables the construction of the vocabulary three times faster.

Table 2 Features of a KTH video. This table shows the average features that a video of the KTH dataset has

Features	
Frames	378
Size	[160 x 120]
Number of scales	4
Frames per second	25

- Proposal $B_{Disof + op_25 + 3 desc}$ construct a vocabulary 3.8 times faster compared to dense trajectories.
- Proposal $C_{OP + op_6 + 3 desc}$ performs 13 times faster than dense trajectories.

6.2 SVM: model generation

While SVM performance is dependent on the parameters used, determining which ones maximize accuracy is typically accomplished by searching in the parameter space. However, because the majority of video datasets are quite difficult to process due to the amount of spatial and temporal information contained in each video, this task can only be approached on low-resolution videos, such as those in the KTH datasets shown in Table 2.

Table 3 SVM model construction. The computation time required to build an SVM model is shown for each of our proposals together with Wang et al. work. All of our proposals perform faster than the dense trajectories

	Time (s)
Wang et al. Farneback + 3 desc	55.28
Our proposal $A_{Disof + op_{25} + 5 desc}$	30.20 (1.8x)
Our proposal $B_{Disof + op_{25} + 3 desc}$	19.13 (2.8x)
Our proposal $C_{OP + op_6 + 3 desc}$	16.68 (3.3x)

Reducing the computation time for the creation of SVM models can help to accomplish this search task; our proposal $A_{Disof + op_{25} + 5 desc}$, $B_{Disof + op_{25} + 3 desc}$, and $C_{OP + op_6 + 3 desc}$ performs the creation of the SVM model 2, 3, and 3.5 times faster than dense trajectories, respectively, as shown in Table 3.

6.3 Extraction, feature encoding, and classification

The pipeline to recognize actions is formed by three steps: trajectories extraction, feature encoding, and action classification. Therefore, each of the steps must be performed as faster as possible if we want to achieve a real-time system. Our results are presented in Table 5 shown that our proposal $Disof + op_{25} + 5 desc$, $B_{Disof + op_{25} + 3 desc}$, and $C_{OP + op_6 + 3 desc}$ improves the performance by a factor of 1.78, 1.78, and 8, respectively.

The computation time for trajectories extraction is normalized by the number of frames, and the computation time for the descriptor construction is by the number of trajectories presented in the video. So, to have a better understanding, we presented in Table 4 the average number of trajectories extracted by each method in four random instances.

Key trajectories extract between 60 and 90% fewer trajectories than dense trajectories, depending on the configuration chosen. This is an important contribution because a lower number of trajectories enables better management of data for both computational time and physical disk storage.

A full comparison of our proposal is shown in Fig. 12 using the average video in KTH number, described in Table 2. Dense trajectories need 120 seconds to process the

Table 4 Trajectories comparison. The number of trajectories generated by each of our proposals is shown together with the work of Wang et al. in random instances. It is possible to observe that our pro-

Method	77 frames	131 frames	103 frames	120 frames
Wang et al.	342	527	248	652
Our proposal A and B	75 (Red.: 78%)	106 (Red.: 69%)	44 (Red.: 82%)	166 (Red.: 74%)
Our proposal $C_{OP + op_6 + 3 desc}$	60 (Red.: 82%)	35 (Red.: 93%)	18 (Red.: 92%)	104 (Red.: 84%)

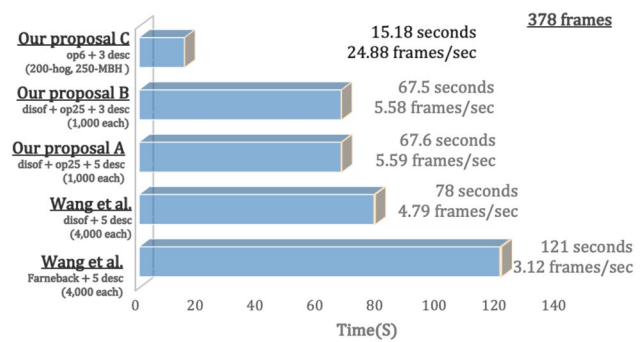


Fig. 12 Computation time: our proposal versus Wang et al. [45]. The computation time needed to process a video of 378 frames for each of our proposals is shown together with the work of Wang et al., which takes almost 2 minutes to process the entire video. The reduction of points, disof and the new size-quantity of descriptors used help us to reduce the needed time by almost half. Replace the optical flow by the subject key points estimation helps to process the video in just 15 seconds, being almost real-time

whole video, but our faster proposal accomplishes this task only in 15 seconds. This means 24.8 frames per second (the KTH framerate is 25 fps) so this configuration runs in real-time in the KTH dataset, as shown in Table 5.

Figure 13 shown that the computation time of key trajectories compared to its dense counterpart at each step is considerably lower. Also, as described in Sect. 7, the accuracy performance is comparable to Wang et al. [45] work. So, the important lessons learned are:

- Dense inverse search reduces the computational time by a factor of 1.5.
- The extraction of trajectories is the most important step to speed-up the recognition task.
- Subject key points estimation can be used as a faster feature tracking method.
- Use subject key points improve the trajectories extraction performance by improving the computation time and reducing the number of trajectories extracted.

posol $C_{OP + op_6 + 3 desc}$ reduced by 82–93% the amount of information that was generated. All videos have a resolution of 160 x 120. A = Disof + op₂₅ + 5 desc, B = Disof + op₂₅ + 3 desc

Table 5 Recognition task: computation time. The computation time required to recognize an action for each of our proposals is shown together with the work of Wang et al.. The time to generate trajectories is normalized concerning the number of frames and of the descriptors by the number of trajectories. Trajectories extraction is the most important step to decrease the computation time. Our proposal $C_{OP + op_6 + 3 desc}$ accelerates this step 8 times. Wang et al. = Farneback + 5 desc, Proposal A = Disof + op_25 + 5 desc, Proposal B = Disof + op_25 + 3 desc, Proposal C = OP + op_6 + 3 desc. the computation time for trajectories extraction is normalized by the number of frames and the computation time for the descriptor constructions by the number of trajectories presented in the video

	Traj. extr (s)	Desc. gen. (s)	Classification (s)
Wang et al.	0.31 (–)	7.61E-05 (–)	0.04 (–)
Our proposal A	0.17 (1.8×)	6.44E-05 (1.18×)	0.01 (4×)
Our proposal B	0.17 (1.8×)	5.77E-05 (1.31×)	0.008 (5×)
Our proposal C	0.037 (8.37×)	4.81E-05 (1.58×)	0.006 (6.6×)

7 Effectiveness assessment

Intelligent video surveillance systems must fulfill two conditions: be accurate and be faster. In Sect. 6, our experimental results show that our proposal is 8 times faster, and this Section aims to assess the effectiveness of our proposal to accomplish the condition described. All experiments were carried out using the KTH, UCF11, UCF50, and HMDB51 datasets.

Table 6 Accuracy comparison. The accuracy of our proposal is shown together with that of Wang et al. [45]. It is important to note that although the method accelerated amazingly, the accuracy was not affected

Approach	KTH (%)	UCF11(%)
Wang et al. [45]	95.65	84.08
Our proposal $A_{Disof + op_25 + 5 desc}$	95.71	84.29
Our proposal $B_{Disof + op_25 + 3 desc}$	94.9	84.43
Our proposal $C_{OP + op_6 + 3 desc}$	94.20	80.66

As shown in Table 6, our proposal $A_{Disof + op_25 + 5 desc}$ and $B_{Disof + op_25 + 3 desc}$, which are two time faster, got the best performance for the KTH and UCF dataset. So, the key trajectories approach is faster and more accurate than dense trajectories. On the other hand, our proposal $C_{OP + op_6 + 3 desc}$ got a comparable accuracy performance to dense trajectories, but its 8 times faster.

Finally, Table 7 compares our proposal with other state-of-the-art approaches in high-difficulty datasets. In this case, we used fisher vector as a feature encoding method. Nevertheless, the computation time is not affected because the major factor reducing the computation time is the trajectories extraction step. Key trajectories outperform dense trajectories in both accuracy and computational time, and their accuracy performance is comparable to other important works in the field. Speaking of action classes, we found that our method performed correctly. However, it may not be the best option in specific situations, for example, in videos where the human body is not shown (like some instances of

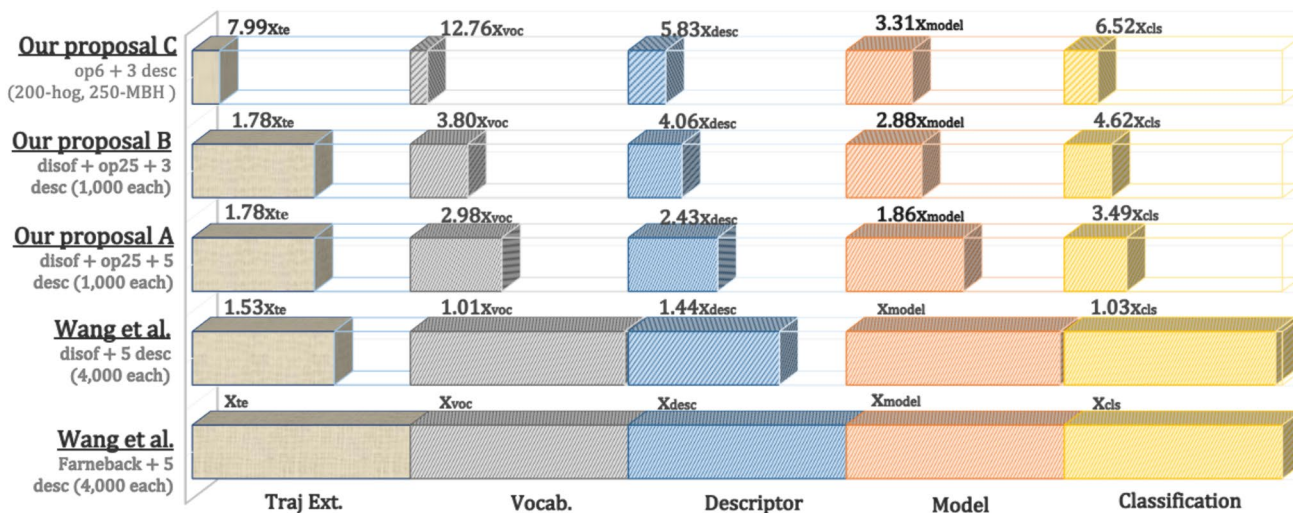


Fig. 13 Speed-up of our proposal by steps. For the trajectory’s extraction, using disof: 1.5 faster. Less number of points 1.78 faster. Key points estimation: 8 times faster. For vocabulary construction, fewer visual words and descriptors: 3.8 faster, using our encoding method:

12 faster. For the descriptor generation, using our procedure is almost six times faster and the classification was improved by a factor of 7

Table 7 Accuracy comparison. Key trajectories outperform Dense Trajectories and had a similar performance to the state-of-the-art methods

Approach	UCF50 (%)	HMDB51(%)
Key Trajectories	92.9	81.3
Wang et al. [45]	91.2	57.2
de Souza et al. [11]	92.5	70.4
Lan et al. [21]	94.4	65.1
Wang et al. [47]	–	82.48

the mixing class of the UCF50 dataset) due to the estimation of subject key points won't be accurate.

8 Conclusions

Understanding what a subject is doing in a video is an important task of computer vision that can be applied to create intelligent video surveillance systems. To work in real-world applications, algorithms must fulfill two conditions: be faster and be accurate.

We focused on analyzing the weaknesses of dense trajectories and solving them through deep learning techniques such as pose estimation. Our proposal, named key trajectories, enables us to recognize actions 8 times faster, which, as far as we know, is the first time that the KTH for the first time, to recognize actions in the KTH in real-time (~ 24 fps) accurately.

Firstly, we focused on the trajectories extraction step, where we found that the dense inverse algorithm as feature tracking makes the action recognition process 1.5 times faster and 2 times faster if we use subject key points for sampling frame locations. Also, we studied the distinctiveness and representativeness of subject key points and found a subset of 6 crucial points that enables us to perform the recognition task 8 times faster with 90% fewer trajectories.

A central contribution of our work is a search on the hyperparameters space of action recognition. We found that the computation time grows is related to the video descriptor length. So, we examined the distinctiveness of each descriptor and discovered that using a higher number of visual words to generate the video representation does not impact on the classification accuracy, but it seriously affects the computation time. Then, we observed that it is recommendable to reduce this number from 4000 to 1000 (~ 3.5 times faster). To further reduce the size of the vectors, we introduced a new technique for encoding the trajectories that helps us build a vocabulary (~ 12 times faster) whose size only requires a 5% of the Wang et al. approach [45]. In the same way, we contributed an analysis of the parameter space of the SVM classifier for the action recognition task,

finding that the classification results have its highest values for values of gamma equal to 0.001 and values of C higher than 100.

8.1 Future work

A significant difficulty with dense trajectories is that feature encoding does not account for the trajectories' time-space relationships. As a result, dense trajectories are incapable of fully representing an action. As a future project, we propose to investigate several methods for including time-space relationships in order to improve action detection accuracy. Also, we will analyze our method with other feature encoding methods and explore other datasets that include RGB-D data.

Another area for improvement is the trajectory extraction step, as it currently uses a predetermined trajectory length. However, because the temporal information associated with actions may be rather complicated, creating methods for comprehending their dynamics might yield more accurate performance.

Acknowledgements F. Camarena gratefully acknowledges the scholarship no. 815917 from CONACyT to pursue his postgraduate studies. The scholarship had no role in study design, data collection and analysis, decision to publish, or preparation of the manuscript.

Funding No funding was received.

Declarations

Conflict of interest The author declares that he has no conflict of interest.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons licence, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons licence, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons licence and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this licence, visit <http://creativecommons.org/licenses/by/4.0/>.

References

1. Bay H, Ess A, Tuytelaars T, Van Gool L (2008) Speeded-up robust features (surf). *Comput Vis Image Underst* 110(3):346–359
2. Berlin SJ, John M (2016) Human interaction recognition through deep learning network. In: 2016 IEEE international Carnahan conference on security technology (ICCST), pp 1–4. IEEE
3. Blank M, Gorelick L, Shechtman E, Irani M, Basri R (2005) Actions as space-time shapes. In: Tenth IEEE international

- conference on computer vision (ICCV'05) Volume 1, vol 2, pp 1395–1402. IEEE
4. Blunsom P (2004) Hidden markov models. *Lect Notes* August 15(18–19):48
 5. Camiña JB, Medina-Pérez MA, Monroy R, Loyola-González O, Villanueva LAP, Gurrola LCG (2018) Bagging-randomminer: a one-class classifier for file access-based masquerade detection. *Mach Vis Appl*. <https://doi.org/10.1007/s00138-018-0957-4>
 6. Cao Z, Simon T, Wei SE, Sheikh Y (2017) Realtime multi-person 2d pose estimation using part affinity fields. In: *CVPR*
 7. Chang L, Pérez-Suárez A, Hernández-Palancar J, Arias-Estrada M, Sucar LE (2017) Improving visual vocabularies: a more discriminative, representative and compact bag of visual words. *Informatica* 41(3)
 8. Chéron G, Laptev I, Schmid C (2015) P-cnn: pose-based cnn features for action recognition. In: *Proceedings of the IEEE international conference on computer vision*, pp 3218–3226
 9. Dalal N, Triggs B (2005) Histograms of oriented gradients for human detection. In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, vol 1, pp 886–893. IEEE
 10. Dalal N, Triggs B, Schmid C (2006) Human detection using oriented histograms of flow and appearance. In: *European conference on computer vision*. Springer, pp 428–441
 11. De Souza CR, Gaidon A, Vig E, López AM (2016) Sympathy for the details: dense trajectories and hybrid classification architectures for action recognition. In: *European conference on computer vision*. Springer, pp 697–716
 12. Du Y, Chen F, Xu W, Li Y (2006) Recognizing interaction activities using dynamic bayesian network. In: *18th international conference on pattern recognition (ICPR'06)*, vol 1, pp 618–621. IEEE
 13. Du Y, Wang W, Wang L (2015) Hierarchical recurrent neural network for skeleton based action recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 1110–1118
 14. Harris CG, Stephens M, et al (1988) A combined corner and edge detector. In: *Alvey vision conference*, vol 15, pp 10–5244. Citeseer
 15. Herath S, Harandi M, Porikli F (2017) Going deeper into action recognition: a survey. *Image Vis Comput* 60:4–21
 16. Huang Z, Wan C, Probst T, Van Gool L (2017) Deep learning on lie groups for skeleton-based action recognition. In: *Proceedings of the 2017 IEEE conference on computer vision and pattern recognition (CVPR)*, pp 1243–1252. IEEE computer Society
 17. Jégou H, Douze M, Schmid C, Pérez P (2010) Aggregating local descriptors into a compact image representation. In: *CVPR 2010-23rd IEEE conference on computer vision & pattern recognition*, pp 3304–3311. IEEE Computer Society
 18. Jhuang H, Gall J, Zuffi S, Schmid C, Black MJ (2013) Towards understanding action recognition. In: *2013 IEEE international conference on computer vision*, pp 3192–3199. <https://doi.org/10.1109/ICCV.2013.396>
 19. Kroeger T, Timofte R, Dai D, Van Gool L (2016) Fast optical flow using dense inverse search. In: *European conference on computer vision*. Springer, pp 471–488
 20. Kuehne H, Jhuang H, Garrote E, Poggio T, Serre T (2011) Hmdb: a large video database for human motion recognition. In: *2011 international conference on computer vision*, pp 2556–2563. IEEE
 21. Lan Z, Lin M, Li X, Hauptmann AG, Raj B (2015) Beyond gaussian pyramid: multi-skip feature stacking for action recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 204–212
 22. Laptev I (2005) On space-time interest points. *Int J Comput Vis* 64(2–3):107–123
 23. Li Y, Li W, Mahadevan V, Vasconcelos N (2016) Vlad3: encoding dynamics of deep features for action recognition. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 1951–1960
 24. Liu J, Luo J, Shah M (2009) Recognizing realistic actions from videos “in the wild”. In: *2009 IEEE conference on computer vision and pattern recognition (CVPR)*, pp 1996–2003. IEEE
 25. Lowe DG (2004) Distinctive image features from scale-invariant keypoints. *Int J Comput Vis* 60(2):91–110
 26. Lowe DG et al (1999) Object recognition from local scale-invariant features. In: *iccv*, vol 99, pp 1150–1157
 27. Luo W, Yang B, Urtasun R (2018) Fast and furious: real time end-to-end 3d detection, tracking and motion forecasting with a single convolutional net. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp 3569–3577
 28. Marszałek M, Laptev I, Schmid C (2009) Actions in context. In: *2009 IEEE conference on computer vision and pattern recognition (CVPR)*, pp 2929–2936. IEEE
 29. Mo L, Li F, Zhu Y, Huang A (2016) Human physical activity recognition based on computer vision with deep learning model. In: *2016 IEEE international instrumentation and measurement technology conference proceedings (I2MTC)*, pp 1–6. IEEE
 30. Ng AY, Jordan MI (2002) On discriminative vs. generative classifiers: a comparison of logistic regression and naive bayes. In: *Advances in neural information processing systems*, pp 841–848
 31. Peng X, Zou C, Qiao Y, Peng Q (2014) Action recognition with stacked fisher vectors. In: *European conference on computer vision*. Springer, pp 581–595
 32. Perronnin F, Sánchez J, Mensink T (2010) Improving the fisher kernel for large-scale image classification. In: *European conference on computer vision*. Springer, pp 143–156
 33. Reddy KK, Shah M (2013) Recognizing 50 human action categories of web videos. *Mach Vis Appl* 24(5):971–981
 34. Rodríguez J, Medina-Pérez MA, Gutierrez-Rodríguez AE, Monroy R, Terashima-Marín H (2018) Cluster validation using an ensemble of supervised classifiers. *Knowl Based Syst* 145:134–144. <https://doi.org/10.1016/j.knosys.2018.01.010>. <http://www.sciencedirect.com/science/article/pii/S0950705118300091>
 35. Schuldt C, Laptev I, Caputo B (2004) Recognizing human actions: a local svm approach. In: *Proceedings of the 17th international conference on pattern recognition, 2004. ICPR 2004*, vol 3, pp 32–36. IEEE
 36. Scovanner P, Ali S, Shah M (2007) A 3-dimensional sift descriptor and its application to action recognition. In: *Proceedings of the 15th ACM international conference on Multimedia*, pp 357–360. ACM
 37. Shi Y, Tian Y, Wang Y, Huang T (2017) Sequential deep trajectory descriptor for action recognition with three-stream CNN. *IEEE Trans Multimed* 19(7):1510–1520
 38. Simon T, Joo H, Matthews I, Sheikh Y (2017) Hand keypoint detection in single images using multiview bootstrapping. In: *CVPR*
 39. Simonyan K, Zisserman A (2014) Two-stream convolutional networks for action recognition in videos. In: *Advances in neural information processing systems*, pp 568–576
 40. Sivic J, Zisserman A (2003) Video google: a text retrieval approach to object matching in videos. In: *null*, p 1470. IEEE
 41. Soomro K, Zamir AR, Shah M (2012) Ucf101: a dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*
 42. Vail DL, Veloso MM, Lafferty JD (2007) Conditional random fields for activity recognition. In: *Proceedings of the 6th international joint conference on Autonomous agents and multiagent systems*, p 235. ACM

43. Veeriah V, Zhuang N, Qi GJ (2015) Differential recurrent neural networks for action recognition. In: Proceedings of the IEEE international conference on computer vision, pp 4041–4049
44. Wang H, Kläser A, Schmid C, Liu CL (2011) Action recognition by dense trajectories. In: 2011 IEEE conference on computer vision and pattern recognition (CVPR), pp 3169–3176. IEEE
45. Wang H, Kläser A, Schmid C, Liu CL (2013) Dense trajectories and motion boundary descriptors for action recognition. *Int J Comput Vis* 103(1):60–79
46. Wang H, Schmid C (2013) Action recognition with improved trajectories. In: Proceedings of the IEEE international conference on computer vision, pp 3551–3558
47. Wang L, Koniusz P, Huynh D (2019) Hallucinating idt descriptors and i3d optical flow features for action recognition with cnns. In: Proceedings of the 2019 international conference on computer vision. IEEE, Institute of Electrical and Electronics Engineers
48. Wang L, Qiao Y, Tang X (2015) Action recognition with trajectory-pooled deep-convolutional descriptors. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4305–4314
49. Wei SE, Ramakrishna V, Kanade T, Sheikh Y (2016) Convolutional pose machines. In: CVPR
50. Willems G, Tuytelaars T, Van Gool L (2008) An efficient dense and scale-invariant spatio-temporal interest point detector. In: European conference on computer vision. Springer, pp 650–663
51. Zhang B, Wang L, Wang Z, Qiao Y, Wang H (2016) Real-time action recognition with enhanced motion vector CNNs. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2718–2726
52. Zhang S, Wei Z, Nie J, Huang L, Wang S, Li Z (2017) A review on human activity recognition using vision-based method. *J Healthc Eng* 2017
53. Zhou X, Yu K, Zhang T, Huang TS (2010) Image classification using super-vector coding of local image descriptors. In: European conference on computer vision. Springer, pp 141–154

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.