Tech United Eindhoven Team Description 2015

Cesar Lopez, Ferry Schoenmakers, Koen Meessen, Yanick Douven, Harrie van de Loo, Dennis Bruijnen, Wouter Aangenent, Bob van Ninhuijs, Matthias Briegel, Patrick van Brakel, Frank Botden, Robin Soetens, Tom Albers, Jan Romme, Stefan Heijmans, Camiel Beeren, Marjon van 't Klooster, Remco Oudshoorn, Lotte de Koning, René van de Molengraft

Eindhoven University of Technology,
Den Dolech 2, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
www.techunited.nl, techunited@tue.nl

Abstract. In this paper we discuss the progress in mechanical, electrical and software design of our middle-size league robots over the past year. Recent process in software includes intercepting a lob pass and improvement of 3D ball detection by combining omnivision output of different robots. For better localization the distance mapping and mapping center are determined during the game to compensate for tilt variations. To improve the acceleration capabilities and agility of the three-wheeled robot an a-symmetric trajectory planner is developed and implemented. Human-robot interaction is this year further explored by using hand gestures to coach our robots. Finally a simulator link is developed to run a simulation with multiple computers which makes it possible to simulate matches of two full teams competing against each other.

Keywords: RoboCup Soccer, Middle-Size League, Multi-Agent Coordination, Trajectory Planning, Communication Interface, Human-Robot interaction, Calibration

1 Introduction

Tech United Eindhoven is the RoboCup team of Eindhoven University of Technology. Our team consists of PhD, MSc and BSc students, supplemented with academic staff members from different departments. The team was founded in 2005 and originally only participating in the Middle-Size League (MSL). Six years later service robot AMIGO was added to the team, which participates in the RoboCup@Home league. Knowledge acquired during the design of our soccer robots proved to be a valuable resource in creating a service robot.

This paper describes our major scientific improvements over the past year. It is a part of the qualification package for the RoboCup 2015 World Championships in China and contains six main sections. First we introduce shortly our current robot platform. Next, Section 3 presents our improved (3D) ball perception. This improvement is required to enable our next new feature, lob-passes described in Section 4. Section 5 discusses improvements in our localization algorithm. It presents an adaptation method to correct the calibration for tilt of the robot and tilt of the mirror with respect to the turtle. An improved trajectory planner with a-symmetric acceleration and deceleration limits is described in Section 6. In Section 7 a new coaching method based on hand gestures is presented and the final section presents a simulator link, which makes it possible to run one simulation using multiple computers.

2 Robot Platform

Our robots have been named TURTLEs (acronym for Tech United RoboCup Team: Limited Edition). Currently we are employing the fifth redesign of these robots, built in 2009, together with a goalkeeper robot which was built one year later (Fig. 1). Development of these robots started in 2005. During tournaments and demonstrations, this generation of soccer robots has proven to be very robust. The schematic representation published in an earlier team description paper [5] still cover the main outline of our robot design.

Changes regarding the robot platform, the 12V Maxon motors will this year be driven by Elmec Violin 15/60 amplifiers and two Lithium Polymer batteries [2]. A detailed list of hardware specifications, along with CAD files of the base, upper-body, ball handling and shooting mechanism, has been published on a ROP wiki [1]. Our qualification page contains a detailed overview of the mechanical and electrical hardware design and the software architecture. [1].

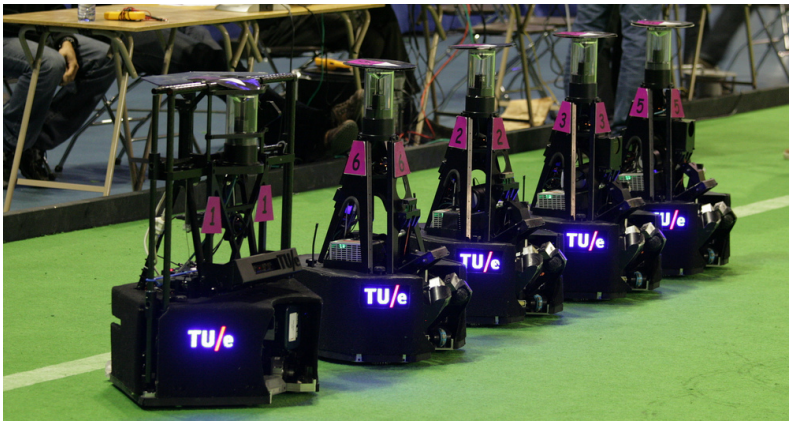


Fig. 1. Fifth generation TURTLE robots, with on the left the goalkeeper robot.

3 Ball perception

Last year a kinect sensor has been added to all robots to observe balls which are moving through the air. This sensor has proven its value when used by the keeper to detect the lob ball shot at our goal. However, there are some limitations with the kinect sensors when placed on the field players. For example, the kinect sensor has a limited view angle compared to for example omnivision. Therefore, the omnivision remains essential for ball detection. However, our current omninvision algorithm is heavily distorted by ball which are not on the ground. This section presents an improved ball detection method using omnivision for balls moving through free air. The balls found using this algorithm can be easily combined with the ball features detected by the kinect, which is further explained in Section 3.1.

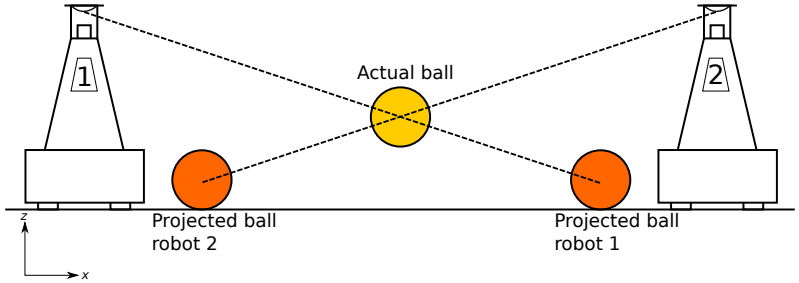


Fig. 2. The intersection point of the vision lines of each robot represents the actual ball position.

The current omnivision algorithm projects the observed ball on the field to obtain its x and y coordinate. This results in an offset if the ball is moving through the air. A possible solution to estimate this offset and to include the z coordinate of the ball is to use triangulation. To triangulate the 3D position of the ball, at least two robots with vision on the ball are needed. The vision lines

[1] http://techunited.nl/en/turtle-qualification

of the robots and their projected balls are known (Figure 2). If we know at least two vision lines, an intersect point of all lines can be calculated. The actual ball should be on the intersection point of those lines. And therefore, the actual ball position (x, y, z) can be calculated which is needed to enable lob-passes, presented in the next section.

3.1 Ball model

A ball model is used to combine measurements from different sensors and to extract relevant information from the ball position measurements. A hypothesis-tree based sequential clustering algorithm [6] is adopted, with some adaptations to make it run in real-time and to deal with the ball dynamics. An estimate of the current ball state is returned. Figure 3 shows an x, z plot of the output of the ball model based on simulated ball measurements. With the new omnivision algorithm, the input of the ball mode is improved and ball features from both kinect and omnivision can easily be combined to obtain an accurate 3D ball position.

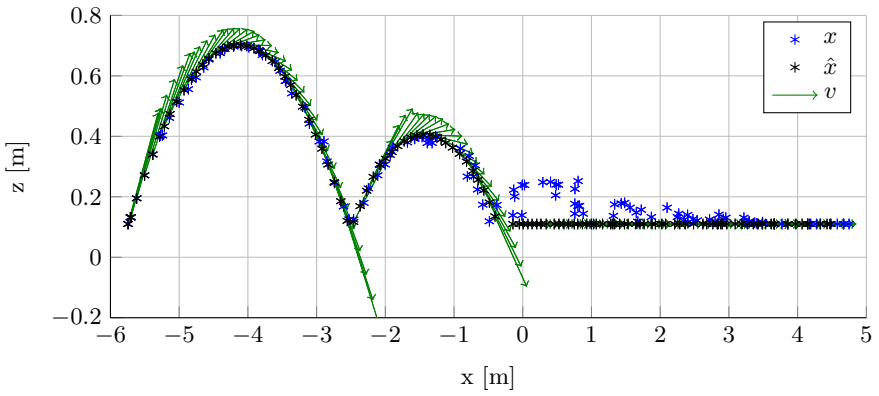


Fig. 3. Ball state estimate for a simulated lob pass with noise. x is measured ball position, $[\hat{x}, v]$ is estimated ball state.

4 Lob passes

Another improvement of this year is the introduction of the lob pass. With the improved 3D ball tracking discussed in the previous section, the retrieved ball information is accurate enough to intercept passes through free air. The pass-giver chooses a desired pass target, based on the current game-play situation and several ball trajectory predictions. When the ball is shot, the output of the ball model is used to adjust the point of intercept for the shooting inaccuracy. In this section, the ball model and intercept strategy for a lob pass are discussed.

4.1 Intercept strategy

The receiving robot corrects for the shooting inaccuracy using a set of predicted bounce locations. The most favourable interception point is the current ball-tracking based prediction, derived from measured ball positions during the lob pass. The least favourable interception point is the initial estimate, as derived by the pass-giver before shooting. Figure 4 shows a flowchart-diagram of the intercept strategy. First, several validation conditions are evaluated. One of these conditions is an accuracy constraint. Due to noise on the ball measurements, the ball-tracking based predictions can be modeled as a bivariate distribution. The accuracy is derived from the sample covariance on a buffer of predicted locations. For all valid predictions, an intercept confidence rating is determined based on the estimated time of arrival and the magnitude and velocity of the incoming ball vector at intercept.

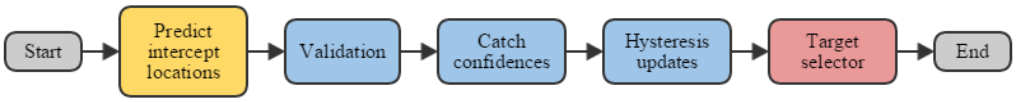


Fig. 4. Basic outline intercept strategy.

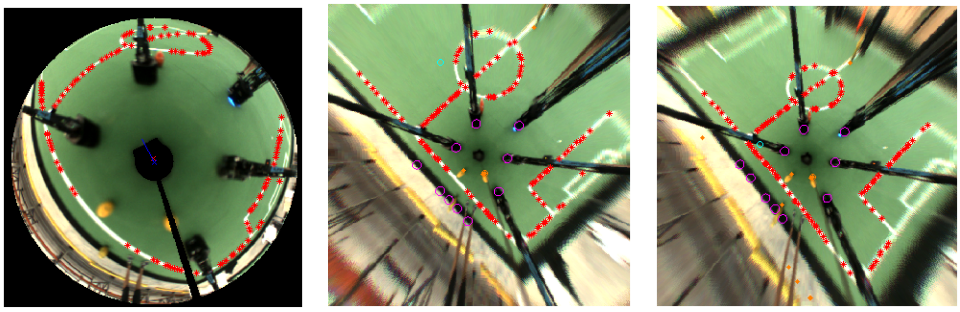
Each prediction receives a hysteresis update. Finally, a target selector chooses a move-target, derived from the prediction with the highest hysteresis count. A move-target is different from an intercept target due to the ball-to-robot distance and an extra 'driving-forward'-distance. The latter is added to the intercept strategy to create a forward robot velocity at intercept. This increases the catch rate for the current ball handling system, which is designed to catch flat passes rather than lob passes. The lob pass is tested in experiments. The robot trajectory is similar to what is expected from simulation. To improve the catch rate further, the ball handling system should be extended for lob pass intercept.

5 Adaptation of localization parameters

For localization, line points in the camera image are being mapped onto an a priori known field geometry. The left image in Figure 5 shows such a camera image. As explained in [3], this line points mapping is parameterized with 9 parameters:

- Distance mapping from pixels to meters (2 parameters)
- Mapping center (2 parameters)
- Robot center (2 parameters)
- Robot pose (x,y,φ) (3 parameters)

These parameters are calibrated offline with a number of images at different locations using global optimization. By separating the mapping center and the robot center, small tilts of the camera and/or mirror can be compensated for. If the mapping parameters are calibrated well, a mapped image is obtained as shown in the middle image of Figure 5. If the parameters are not calibrated well, the right image is obtained. A similar distortion is obtained when the robot is tilted, hence, this provides a possibility to compensate for tilts.



(a) Camera image with line detection

(b) Camera image to field mapping with correctly calibrated parameters

(c) Camera image to field mapping with shifted vertical offset which is also observed when robot tilt occurs

Fig. 5. Camera images showing line detection.

Up to the year 2013, only the pose was being optimized during robot operation. The remaining parameters were fixed. Localization became difficult in case that tilt occurs during a game. To

compensate for tilt variations, from the year 2014, the distance mapping and mapping center are also determined during a game. Hence, seven parameters are continuously being optimized for each new image. This has shown to work well, tilts up to 2 degrees can be easily coped with. This is comparable to lifting one wheel by about 1 cm.

The computational load of the distance mapping had to be optimized to make this additional optimization possible in real-time. Images come at a rate of 50 Hz. Each image, 2 Nelder-Mead simplex optimizations with maximally 300 cost function evaluations are run. Furthermore, for each evaluation maximally 150 line points are being mapped. In total, about 5e6 line points have to be mapped each second. The remaining problem is chatter in both the ball and obstacles detection if localization is lost. This happens occasionally e.g. due to objects that block the view. As the localization algorithm fails, the default mapping parameters are being used resulting in a transient in both the ball and the obstacle position predictions. To reduce this disturbance, slow adaptation of the mapping parameters is proposed. This will reduce ball/obstacle chatter when localization is lost resulting in better ball intercepts in a dynamic environment with multiple obstacles.

6 A-symmetric trajectory planner

One of the major drawbacks of the three-wheeled robot of Tech United is its orientation dependent acceleration capability. This is caused by the angle between the wheels illustrated in Figure 6a. When the robot drives forwards or backwards, i.e. in the y-direction, wheel one (w_1) and two (w_2) are active while wheel three (w_3) does not provide torque/force. If the robot drives in the x-direction or rotates, all three wheels are active and providing torque/force.

During a RoboCup game, the robot is mainly driving in the positive y-direction towards or with the ball. During defensive actions or positioning to receive a pass, the robot will also driving in the x-direction. When the robot accelerates in the forward (positive y-) direction, the normal force of the front-wheels is reduced due to backwards tilt of the robot. In case that the robot decelerates, when driving in the positive y-direction, the normal force on the front-wheels is increased caused by the inverse tilt. The friction force between the wheels and the ground is proportional to the normal force of the robot to the ground surface. This friction between the wheels and robot determines the maximum acceleration of the robot without having slip. Hence, the maximum acceleration value is theoretically lower than the maximum deceleration value when driving forwards.

Ideally, the trajectory planner should take into account the wheel configuration, the center of gravity and the grip of the omni-wheels to minimize the slip and maximize the acceleration and deceleration. An a-symmetric trajectory planner is proposed as a first step towards a more effective trajectory-planner. This trajectory-planner uses different limits for the acceleration and the deceleration phase of a trajectory. The resulting trajectories are faster because the robot will drive for a longer time at its maximum speed and the robot will be more agile because it can decelerate faster when its target changes while driving.

The trajectory-planner is implemented as a 2-DoF planner for x, y and a 1-DoF planner for ϕ. This means that the acceleration limit is treated as a 2-DoF constraint for x and y, i.e., $a_x^2 + a_y^2 \leq a_{\max}^2$. An optional flag can be used to reduce the acceleration of ϕ such that the duration of the ϕ-trajectory is equal to the xy-trajectory. This reduces the slip probability in the situation that the trajectory duration in xy is longer than the trajectory duration in ϕ. The output of the planner is a second order position profile (discontinuous acceleration) which is used as a position setpoint for the position controller of the robot.

Figure 6b illustrates an example of the effect of the a-symmetry in the trajectory. The robot is moving towards position B at full speed (3 ms^{-1}) and decides at position A to change its target from position B to position C. The solid line illustrates the fastest trajectory to C if the maximum acceleration and deceleration are equal (1.5 ms^{-2}). The black dashed line (- - -) is the resulting trajectory if the deceleration is set to 150% of the acceleration. The red dotted line (-·-·-) is the result of a deceleration set to 200% of the acceleration. It can be seen that the resulting trajectory is much faster. The deceleration to acceleration ratio will be tuned and tested in several situations to make the robot more agile during the game.

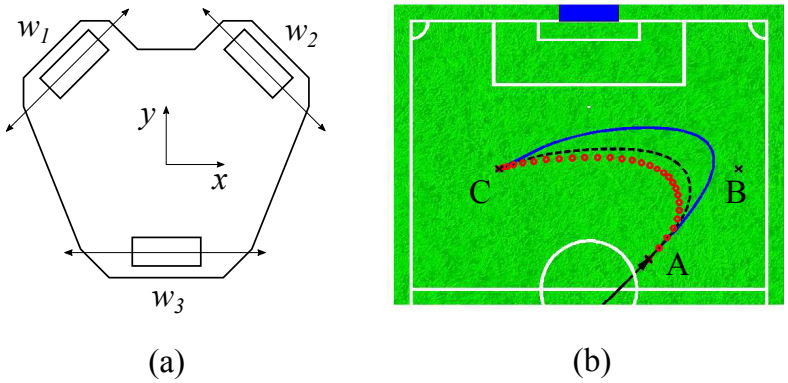


Fig. 6. (a) Techunited robot wheel configuration. The positive y-direction is the forward direction. (b) Example trajectory where the robot is moving towards position B at full speed (3 ms^{-1}) and decides at position A to change its target from position B to position C. Blue solid line: deceleration = acceleration, black dashed line: deceleration = 1.5*acceleration, red dotted line: deceleration = 2*acceleration.

7 Human coaching

For the previous RoboCup MSL, the human-robot coaching was introduced making it possible to instruct a robot with high-level instructions like 'shoot more often' or to change tactics, for example how to take a free kick. To stimulate innovation, coaching by means of qr-codes is no longer permitted at this years RoboCup. As a result, new ways of communication with the robots is needed, leading to the introduction of hand gesture recognition by using the kinect cameras.

The principle behind the recognition algorithm is based on a dedicated geometric descriptor, where a convex hull and its convexity defects are used to detect the shape of the hand [4]. Using the openCV library one can easily extract this information out of the incoming images. While this concept has been proven to work almost flawless, some difficulties have been encountered during the implementation and experimentation of it, especially with the recognition rate in different lighting conditions. As a result, a more robust concept will be developed, based on the seven Hu invariant moments on a binary image of the hand [7].

In order to achieve this, the incoming images first need to be filtered and converted to a binary image, to ensure a faster and more accurate detection of the hands. Hereto, two major steps are performed, i.e. the filtering based on color and on depth. Filtering on depth implies losing all the pixels (set them to black or 0 in the binary image) that are too far away (over 5 meters) or too close by (less than 50 cm). The filtering on color implies that every pixel that has a color within a predefined range is set to 1 in the binary image, while all other pixels are set to 0. When done correctly, a binary image showing only the wanted hand to recognize will be the result. However, a well known problem with hand gesture recognition is the color of the hands, which is extremely difficult to detect, especially in varying lighting conditions. To solve this, a glove with a distinctive color will be used, however an alternative could be working in the YC_bC_r color space.

Once this binary image has been obtained, the Hu moments can be compared to the Hu moments of known gestures stored in a database with the values resulted from many experiments. The goal of this new approach is to have at least five different hand gestures to be recognized with a 80% recognition rate, however testing should still confirm this.

8 Simulator link

The Tech United simulator enables RoboCup developers to simulate the behaviour of the Tech United soccer robots without the need to run their program on actual robots. Therefore, the

development process gets expedited considerably. The simulator runs the same processes that are used on the actual robots with the exception that all modules interacting with the hardware have been replaced by modules that imitate the robots' behaviour. Furthermore, the behaviour of the ball is simulated using an advanced physical model.

By only simulating the behaviour of the low-level hardware, the simulator becomes highly representative of the behaviour of the robots on the field. One of the drawbacks of this approach is that it is computationally expensive because the software of several robots, which usually runs on the separate embedded PCs of the robots, is now run on a single computer. As a consequence, the number of robots that can actually be simulated is limited. In order to increase this number, the so-called 'Simulator Link' has been developed.

It is a communication interface, which makes it possible to connect two Tech United simulators that run on different computers, and thereby makes it possible to simulate matches of two teams competing against each other. Every team is simulated on a separate computer, and the 'Simulator Link' communicates the data that needs to be exchanged, like the ball and robot positions, between the computers. This is done by adding a communication module to the simulator, which handles the communication between computers and the different processes running on the same computer. An illustration of the communication architecture can be seen in Figure 7.

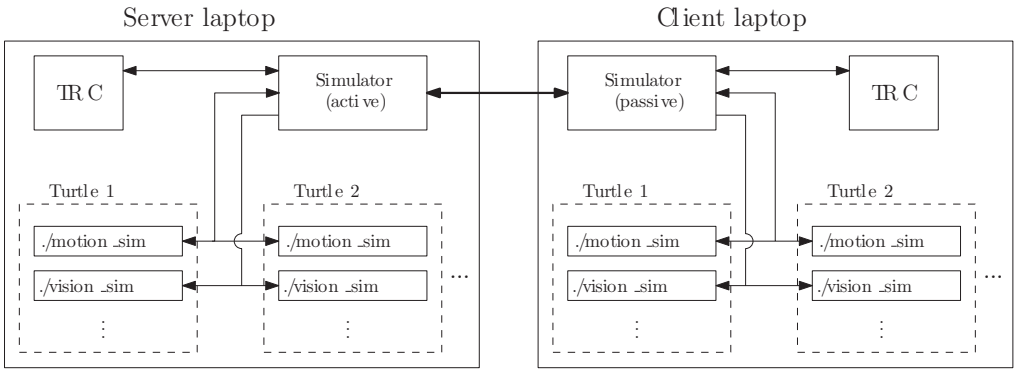


Fig. 7. Communication structure between a server and a client laptop both simulating a number of robots.

To ensure a high enough data transfer rate, the communication between the two simulators is done using a UDP connection (strong line in the picture). The interprocess communication is handled by mutex-protected shared memory structures (thin line). The data packages that are exchanged between the simulators contain, besides the ball and robot positions, global robot settings, like team color and activity status, the refbox commands, and shooting forces applied by each robot. The input data is gathered from every simulated robot's 'motion_sim' process. On the other computer, the robot positions are then translated to obstacle positions and are then fed to the 'vision_sim' process. The same happens with the ball positions. Refbox commands are sent from the simulator process to the 'motion_sim', where they are used as input to behavioural logic.

To prevent that the two simulators calculate different ball behaviours, it was chosen to only use one of the simulators for these calculations, the 'Server' simulator. The other simulator's ball position calculations are circumvented by using the communicated ball position as direct input. The second simulator process, the passive 'Client', is only run to make use of its communication infrastructure so that it can distribute the data packages among the different simulated robots. Furthermore, Refbox commands can only be given by the 'Server' simulator in order to avoid conflicting commands being issued.

Because the 'Simulator Link' makes it possible to simulate two teams playing against each other, it paves the way for high-level learning being applied to the logic that determines a team's

strategy. One can imagine a scenario where two teams play against each other, while a learning algorithm continuously adjusts the strategies. It is also possible to replace one team's logic by a model of an opponent's team that was obtained using measurement data from previous matches against this team. This way, the own strategy can be optimized autonomously using a learning algorithm by simulating a high number of matches.

9 Conclusions

In our team description paper we have discussed concrete steps towards more accurate ball position estimation which enables intercepting lob balls. Furthermore the localization software is improved by compensating for small tilt of the camera during game-play. An a-symmetric trajectory planner is implemented to improve the acceleration capabilities and agility of the robots. Finally we have developed hand gesture based human-coaching software and a simulator link which allows to run one simulation using multiple computers. Altogether we hope our progress contributes to an even higher level of dynamic an scientifically challenging robot soccer during RoboCup 2015 in China, while at the same time maintaining the attractiveness of our competition for a general audience.

References

1. Robotic open platform. `http://www.roboticopenplatform.org/wiki/TURTLE`.
2. Specification lithium polymer batteries. `http://kypom.com/cer_detail.asp?id=27`, `http://kypom.com/cer_detail.asp?id=31`.
3. Dennis Bruijnen, Wouter Aangenent, Jeroen van Helvoort, and René van de Molengraft. From Vision to Realtime Motion Control for the RoboCup Domain. In *IEEE International Conference on Control Applications*, pages 545–550, Singapore, 2007.
4. Bruno Emile Jean-Francois Collumeau, Remy Leconge and Helene Laurent. Hand gesture recognition using a dedicated geometric descriptor. *Image Processing Theory, Tools and Applications (IPTA),3rd International Conference on*, 2012.
5. Tech United Eindhoven MSL. Tech united eindhoven team description 2014, 2014.
6. H.J. Schubert. Sequential clustering with particle filters - Estimating the number of clusters from data. *Proc. 8th Intern. Conference on Information Fusion*, 2005.
7. Yanmin Yin Yun Liu and Shujun Zhang. Hand Gesture Recognition Based on HU Moments in Interaction of Virtual Reality. *Intelligent Human-Machine Systems and Cybernetics (IHMSC), 4th International Conference on*, 2012.