Noname manuscript No.
(will be inserted by the editor)

Effective Real-Time Visual Object Detection

Francisco Martín · Manuela Veloso

Received: date / Accepted: date

Abstract Autonomous mobile robots equipped with visual perception aim at detecting objects towards intelligently acting in their environments. Such real-time vision processing continues to offer challenges in terms of getting the object detection algorithm to process images at the frame rate of live video. Our work contributes a novel algorithm that is capable of making use of all the frames, where each frame is efficiently processed as a "continuation" of the processing of the previous frames. From the 2D camera images as captured by the robot, our algorithm, Wave3D, maintains 3D hypotheses of the presence of the objects in the real 3D world relative to the robot. The algorithm does not ignore any new frame and continues its object detection on each frame by projecting the 3D hypotheses back into the 2D images to focus the object detection. We can view Wave3D as validating the 3D hypotheses in each of the images in the live video. Wave3D outperforms the static single-image classical approach in processing effort and detection accuracy, in particular for moving objects. In addition, the resulting reduced vision processing time translates into more computation available for task-related behaviors, as greatly needed in situated autonomous intelligent robot agents. We conduct targeted experiments using the humanoid NAO robot that illustrate the effectiveness of Wave3D.

Keywords Robot Perception · Humanoid · Robot Soccer.

F, Martín
Robotics Lab, GSyC, Universidad Rey Juan Carlos. Móstoles, Madrid, Spain
E-mail: francisco.rico@urjc.es

M. Veloso
Computer Science Department, Carnegie Mellon University, 5000 Forbes Avenue. Pittsburgh, PA, United States
E-mail: mmv@cs.cmu.edu

1 Introduction

Autonomous mobile robots are increasingly being used in our environments to perform concrete complex tasks. Such robot agents need to perform a set of computationally intensive functions, in order to be able to perceive, to reason about, and to act in their surroundings.

The robot's actuation, whether it is to manipulate objects or just to navigate throughout its environment, is based on the information provided by its sensors. The success of this actuation depends on the reliability and accuracy of the sensory information. One of the richest and most complex sensory information sources is the visual information as captured by cameras. Many of the modern robots are equipped with cameras to observe the world and detect the objects relevant to accomplish the specific tasks.

Robots perform their tasks in a closed loop between perception and control (behaviors). Control includes the computation of the next action towards self-localization, navigation, cooperation, or object manipulation. In dynamic environments in particular, the detection of relevant objects to the actuation has to be done in real time, i.e., with no delay in the perception/control loop. As an example, in robot soccer, a moving ball needs to be detected in real time, so that the perception and control loop moves the robot's camera to focus the attention on the ball. Tracking the ball is a challenging coordination problem as the ball and the robot move usually not at low speeds. In order to succeed, the control to actuate the camera's motors should be calculated with the most updated perception information, as made available at the frame rate of the camera (e.g., for a 30Hz frame rate, the available computation time in between frames is 33ms.) If perception processing, and therefore object detection, takes longer than the frame rate, then the control algorithm makes decisions based on old perception (i.e., old positions of the ball), most probably leading to ineffective actuation that may

miss the ball. As another example, mobile robots have to detect people and obstacles in real time, when moving in our daily environments. If the perceptual information (visual or of other type) used for the selection of the navigation action has a big delay, the robot could collide with obstacles, including with humans.

So, object detection has to be performed efficiently by the complete robots, as such robots cannot assign all their computational effort to perceptual processing. Unfortunately, the information extraction from the images in general incurs in intensive and costly processing. Most of the current approaches, as we discuss next in section 2, perform object detection by analyzing highly dimensional static images, which becomes an expensive computational task for a real robot with limited computation resources. Most efforts try to improve this process by reducing the dimension of the images to be analyzed, but they still may not be able to process images at the frame rate of live video, and frames may be discarded, even if they are newer than the ones being processed.

In this work, we contribute a new approach for detecting objects that does not discard any frame in live video, leading therefore the control decisions to be made on the most updated visual information of the world. Our algorithm, Wave3D, achieves this feature by being able to process every new frame as a continuation of the processing of previous frames. The algorithm generates 3D hypotheses of the presence of the objects in the real 3D world relative to the robot. These hypotheses are validated by projecting them to the 2D image coordinates in the images fetched by the robot camera. The interesting aspect is that these 3D hypotheses are independent from a specific image frame, as opposed to the 2D projections that depend on the robot's motors position. Because of this independence, our algorithm makes use of every new image fetched by the camera, which let us to perform the object detection process in live video.

Furthermore, we achieve another improvement by generating 3D hypotheses on the most probable positions based on the previous frames. For example, the hypotheses for ball detection in robot soccer are only generated on the playing floor, starting where the ball was detected previously. The algorithm progresses with its search in a wave-like manner, as a propagation from the point of the hypotheses. The Wave3D algorithm greatly improves the processing effort which, frees computation for task-related behaviors.

2 Related approaches

Computer vision is a classic field in computer science which is focused in extracting information from images for its use in computer applications. There are many applications in computer vision. Works on face and object detection in images extract statistical information, which is compared among them using Neural Networks [1] based or Support Vector Machines [2], among the most common techniques. There are also many application of computer vision technologies for people and traffic surveillance, as [7] and [6]. Techniques for large image processing are becoming more and more popular with the development of works like [5] and Google Goggles focused on processing and classifying images from the web. The majority of the previously cited works process full images because this is made offline or relying on expensive and powerful computers. There are some techniques for real time video processing which focus on a fast object detection with computational restricted resources. One of the most successful video processing technique is Condensation [3][4][9][10]. This technique is based on maintaining a population of samples which represents a probability distribution for detecting objects in images. This technique does not process every pixel in the image, but the ones where the particles are. This technique has successful results, but the search space is always the image. In [11], a MonteCarlo approach is used for multi-object tracking with several cameras. In this case, the search space is the real world, and the particles are projected to the images captured by several cameras in order to validate them.

But we are working with robots, and these resources are not available or it will consume a lot of them. From initial works on computer vision for robots [12], images are being used as an important sensor for getting information from the environment. In the domain in which we are working, where the computational time is limited and the time requirements are very exigent, the efforts are focused on developing efficient computer vision techniques. The most extended approach for object detection is [13], which consists of processing every pixel in an image. This process includes color segmentation, connecting regions with same color characteristics, extracting information from regions (bounding box, centroid, density) and posterior valid regions merging. This method consumes high computational resources because all the pixels in the image have to be classified. This method has been used widely in the RoboCup domain, using techniques like look-up tables or reducing the resolution to make this method computationally more affordable. The state-of-the-art approaches in this domain are scan-lines based [15][14], which tries to make the image processing more efficient. Instead of processing all the pixels in the image, they only process some rows and columns from the image and take note of the color transitions along these lines for a posterior analysis of these transition points. Most of these work detect the relevant objects of the environment in the image space. After detecting the objects in the image, some of them calculate the tridimensional position of the objects using the motors odometry and some previous knowledge about the object size and position (i.e. the ball is always in the floor).

The radical novelty of the method proposed in this paper with respect to most of the previously presented approaches, is doing the inverse process for detecting the objects. Objects are not pixels, they are real elements that exist in the real world, independently of the fact of being captured in an image. We guess possible object positions in the real world, and we use the image to validate these hypotheses. Making the search space independent from the image allows us to refresh the images as soon as they are fetched, doing a real video processing, instead of an image processing. This allows us to work at higher frequencies and to not mind the image resolutions.

3 Wave3D approach

Our focus is on the RoboCup Standard Platform League, which presents a complex problem. Virtually all the classic problems of mobile robotics have to be solved for making a robot play soccer autonomously: locomotion, perception, self-localization, navigation, coordination or generation of behaviors, including the most representative. The environment where the game is played is very dynamic and challenging, with numerous collisions and occlusions caused by other robots that are in the field.

All elements of the game have a distinctive color: The ball is orange, the field is green, white lines and the goals are blue and yellow, respectively. Because of this, since the robots in the RoboCup Standard Platform League are equipped with a camera, the main sensor is vision. All the elements relevant to the robot are perceived by processing images from the camera.

The main feature of the Standard Platform League is that the robot is the same for all participants. As the hardware can not be modified, most efforts are focused on software development. In addition, the robots are autonomous and all processing must be carried on board. Resources are limited and must be optimized for the implementation of control software in a reasonable time.

In several approaches, the detection of relevant elements is done through an exhaustive analysis of the image. This means examining each of the pixels of an image to find related groups of pixels with similar color and correct size. The process is usually quite costly, in terms of computation time. Moreover, the processing of an image is not usually related to the following image, having to repeat the detection process in each cycle.

This work is focused on visual processing in robots. Perceptual information is used to generate commands for robot actuation. This not only means that information must be accurate and reliable, but must be done in real time for effective actuation. If processing a complete image takes too long, the action commands are generated from information that is too old to be effective in some cases.

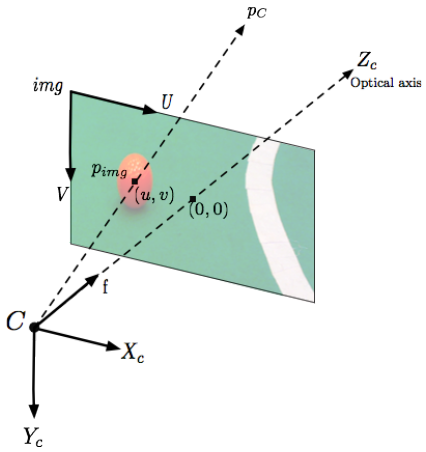

Fig. 1 *Pin-hole* camera model. The camera frame origin C is at the center of the camera. The line connecting the point P_C and the camera center cross the image plane in the point p_{img}, which is its projection in the image.

Our Wave3D algorithm provides a new approach to vision systems for robots. First, the search for the objects is performed in three dimensional space. The system hypothesizes positions where the objects can be found and then validated in the image. This permits to make a real video processing rather than image processing. Thus, the processing time does not depend on the resolution of the images but on the hypothesis to be tested in each cycle. Furthermore, the hypotheses are independent of the images, so the search begun in an image can be continued in the following image. This allows us, on one hand, to yield to the control code and continue later, even using a different image. On the other hand, validation of the hypotheses are conducted using the latest image, even if they change during the detection process.

3.1 Reference frames

Before starting the description of the algorithm Wave3D is necessary to present the reference frames used in the rest of this paper. We define three different reference frames: The image frame, the camera frame and the robot frame. The first is two-dimensional, whereas the latter two are three dimensional.

An image stores the information collected from the camera. An image *pixel* $p_{img}(u, v)$ is the intensity of light or the color stored in the image coordinates (u, v) in the image frame img. The reference frame of the camera C represents points in space with respect to the center of the camera. Figure 1 shows these two reference frames and how they are related using the "Pin-hole" model. The image frame is a plane perpendicular to the axis Z_C, at a distance f far away from the center of the camera. Every point p_C in the camera frame has an unique projection p_{img} in the image plane,

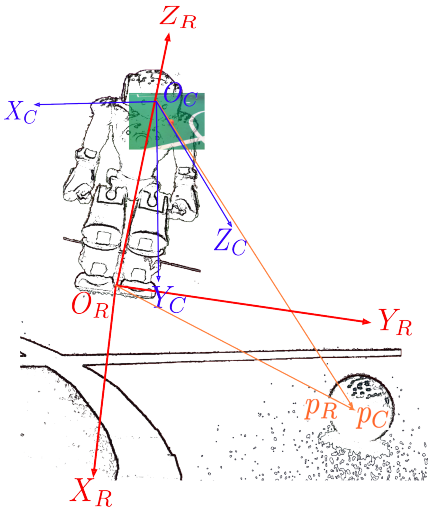

Fig. 2 Robot and camera frames. Projection of the Hypothesis p_C (or p_R with respect the robot frame) in the image.

located at the intersection of the plane with the line linking p_C with the center of the camera.

Instead of using the reference axes of the camera C, we use the reference axes of the robot R. Figure 2 shows O_R as the origin of the reference frame R, which is on the floor between the robot's feets. The points on the axes of the robot are related to its coordinates in the axes of the camera by using the equations 1-3. Each image has an associated matrix RT, which is calculated from the position of the robot's motors. This matrix contains the rotations $R_{a,b}$ and translations T_a (where a and b subindexes are related to any axis) for transforming the points from one reference frame to another one.

$$(p_R^{x,y,z})^T = RT \cdot p_C^{x',y',z'} \tag{1}$$

$$\begin{bmatrix} x_R \\ y_R \\ z_R \\ 1 \end{bmatrix}^T = \begin{bmatrix} R_{x,x} & R_{x,y} & R_{x,z} & T_x \\ R_{y,x} & R_{y,y} & R_{y,z} & T_y \\ R_{z,x} & R_{z,y} & R_{z,z} & T_z \\ 0 & 0 & 0 & 1 \end{bmatrix} = \cdot \begin{bmatrix} x'_C \\ y'_C \\ z'_C \\ 1 \end{bmatrix} \tag{2}$$

$$RT^{-1} \cdot (p_R^{x,y,z})^T = p_C^{x',y',z'} \tag{3}$$

3.2 3D ball hypotheses

The goal of the robot's visual processing is to calculate the position p_R of an object relative to the robot frame R . Instead of performing an image processing to find the object and then transform the position in space of the image img to the space of the robot R, our new approach performs a subject search directly in the space of the robot R. Wave3D

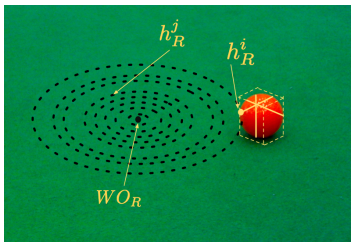

Fig. 3 3D hypothesis generation in wave-like manner centered at WO_R. The hypothesis h_R^j projection corresponds to a green pixel and the search continues. The hypothesis h_R^i projection corresponds to an orange pixel. The bounding box calculated from h_R^i is valid and the search successfully finishes.

generates a set of hypotheses h_R^i in positions where the object can be found. The projection h_{img}^i in the image is calculated from each hypothesis h_C^i. If the coordinate h_{img}^i corresponds to the color of the object sought, it starts a second validation stage. In this second validation stage, we generate a predefined number of points p_{img} next to h_{img}^i to calculate the volume of the detected object. If this volume is consistent with the desired object, the search successfully finishes. If the volume does not correspond to the ball size, the search continues.

The hypotheses are generated in a wave-like manner from an initial position. If the object's position was known before, this will be the initial position of the wave. If the position of the object is unknown, it begins in the center of the robot's field of vision. If an object is detected, subsequent searches begin in the position of the object. In this way, subsequent searches will need to generate a few hypotheses to find the object again. If the wave leaves the whole field of vision of the robot, it restarts again.

Figure 3 is an example of this process for the detection of the ball. The hypotheses begin to be generated from the wave origin WO_R away from it at every step. When the projection of h_R^i in the image plane, h_{img}^i, corresponds to the coordinates of a orange pixel, it begins the second phase of validation. The same figure shows that the bounding box created from h_R^i fits with the ball dimensions, making the search process finish successfully. In the following cycles, the origin of the wave will be the center of the bounding box created from h_R^i.

If the hypotheses are out the field of view of the camera, or is beyond the limits of the field, the search is restarted. Thus, if there were sporadic occlusions, the occluded areas are not permanently discarded. If the ball moves, the search starts from the last known position, regardless of the position of the camera. This provides spatial continuity for the searching, and means that, after checking a few hypotheses, the ball again be found, making efficient the process of tracking.

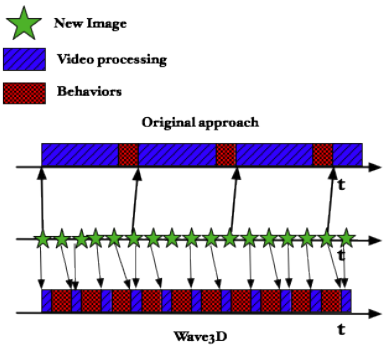

Fig. 4 Execution scheduling in time with the original video module (upper line) and with Wave3D (bottom line). The center line represents the image capture thread.

3.3 Live video processing

The robot should react to changes in their environment properly. To detect these changes, it uses perceptual information obtained in visual processing, which is used to generate the actuation commands of the robot.

The robot software execution is often planned as a closed loop that begins with the processing of images from the camera and ends with the generation of the action commands. Often a robot has to react to visual stimuli in very short periods of time, e.g. to track a ball moving at high speed. The reaction of the robot to track the ball has to be very fast. Any delay in the generation of action commands would result in losing the ball.

Since Wave3D searches for objects in the robot space R, which is independent of the image, it can be used instead of the old one when a new image is available. Figure 4 shows the execution of Wave3D and the original approach. The new images are always used for the detection of objects. For this reason, the actuation commands are more appropriate, as they are generated using the most updated perceptual information. In addition, visual processing can be paused at any time and resume later. Wave3D arbitrarily set a maximum time t_{max} reserved to visual processing. If it exceeds this time, it yields to the execution of behaviors. This will guarantee a minimum rate for the control loop execution. The whole algorithm is described in detail in Algorithm 1.

4 Experiments

Our current approach for visual perception enables a robot Nao to optimize actuation using two simple techniques: increasing the execution frequency of the control loop, and using the most updated image for hypothesis validation. In this section we measure how much the frequency of the control loop is increased using Wave3D with respect to a classical approach, and the improvement provided by this method in the robot actuation.

Algorithm 1: Wave3D algorithm.

```
Ball=getLastBallPosition()
execution_time=0;
start_time = clock();
while execution_time < t_max do
    if camera.isNewImage() then
        image = UpdatedImage();
    h^i_O = getNewHypothesis();
    if h^i_O=0 then
        resetWave();
        continue;
    h^i_C = RobotCoords2CameraCoords(h^i_O);
    h^i_img = ApplyPinHole(h^i_C);
    color = image[h^{x,i}_img][h^{y,i}_img];
    if color == orange then
        BB = calculateBoundingBox(h^i_O);
        if BB.valid() then
            WO_O = BB.center();
            setBall(BB.center());
```

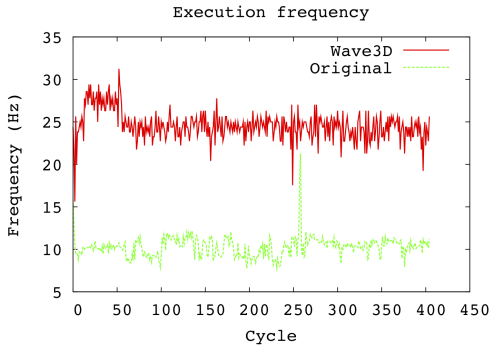

Fig. 5 Frequency of the control loop using the original vision module (dashed line) and using the Wave3D vision module (solid line).

We have designed an experiment to measure the control loop frequency and to compare the time spent by our original vision module, which uses a classical approach, with the Wave3D vision module. In this experiment, the ball is always visible in the image and is moving at different speeds. We set the maximum time t_{max} for video processing in the Wave3D module to 15 ms. Table 1 shows the execution time spent by the original vision module and by the Wave3D vision module. On a few occasions the maximum time slot duration is reached (since the ball is usually found before this threshold), and no other image is available. Figure 5 shows the frequencies of the execution of the original method (10 Hz on average) and Wave3D (24Hz on average).

The benefits of Wave3D in terms of execution speed are evident, but we have yet to demonstrate how this improvement benefits the operation of a Nao robot. The robots must use object detection to self-localize and score goals. It is infeasible to measure the benefits of Wave3D during a soccer match, because there are many confounding factors that can influence the results. Hence, we make the experiments using a basic skill called *TrackingBall* for these experiments.

The goal of the *TrackingBall* skill is to look for the ball and center it in the image at all times. To achieve this goal,

Method	Median	Average	Stdev
Original	43.0 ms	43.305 ms	4.225 ms
Wave3D	6.0 ms	6.306 ms	2.656 ms

Table 1 Computation times for the original method and for Wave3D

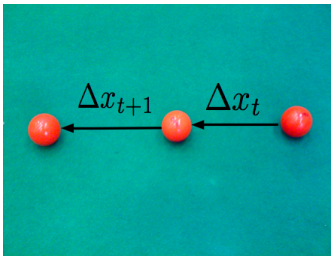

Fig. 6 Sequence of images when the ball is moving. Δx represents the position variation between frames.

the skill takes the ball position provided by the video processing module in the robot frame R and moves the head to point directly at the ball (matching up the Z_C axis and the line that connects the camera with h_C^i, obtained from applying equation 3 to h_R^i, as we showed in Figure 2). This skill is executed in the stripped slots of Figure 4, and sends control commands to the robot neck motors according to the information provided by the vision module.

In Figure 6 we have represented the value that we want to measure. This value is designated Δx or *scan distance* and represents the variation in the position of the ball, in the robot frame, used by the control code to track the ball. If Δx is high (upper image in figure 6), the distance from the previous point that the camera was pointing at and the new point is high, and the head movement will be less smooth. Besides that, if Δx is high enough to set the ball outside the camera range of vision, the ball will be declared lost, and a searching routine begins. If Δx is low (bottom image in figure 6), the movement will be smooth, and the ball will remain in the robot's field of view. The lower the value of Δx at a similar speed of the ball, the more updated the position of the object to track will be and best performance the robot actuation will be.

In this experiment, we measure the variation in the ball position, Δx, while the robot is tracking it at different ball speeds. We activated the *TrackingBall* skill to make the robot track the ball by moving the head without walking. Once the ball is detected by the robot, we manually moved the ball to make the robot adjust the neck position to center the ball in the image. The ball follows a trajectory parallel to the Y_{robot} axis, 1.5 meters far away from the robot. We made several trials at different ball speeds with both vision modules.

Figures 7 and 8 show the results of this experiment for a similar trial. The speed in both cases is lower than 40 cm/s and the experiment begins and ends with the ball stopped. The variation Δx using the original video module (figure 7) varies while the ball is moving, reaching the distance of 25

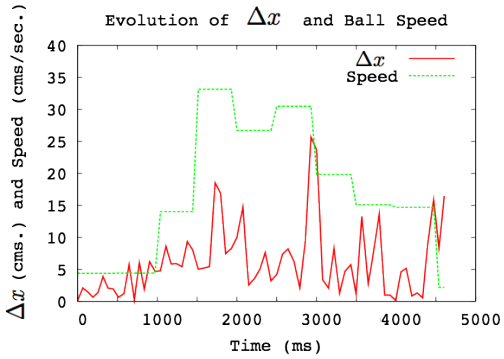

Fig. 7 Variation of Δx (solid line) while the ball is moving at a speed (dashed line) using the original video module.

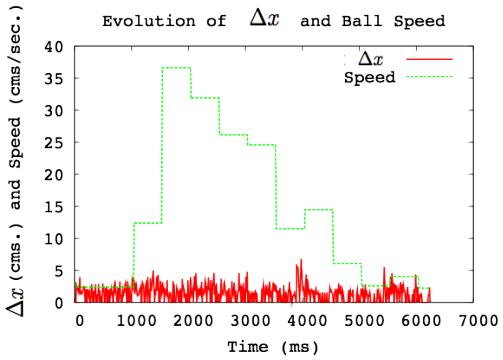

Fig. 8 Variation of Δx (solid line) while the ball is moving at a speed (dashed line) using Wave3D.

cm/s. The variation Δx using Wave3D (figure 8) is always low, with values lower than 5 cm/s. The head movement of the robot is smoother with Wave3D, because of the higher detection frequency, than with the original video module, and the risk of losing the ball is lower as well.

The experiment was conducted at different ball speeds. Figure 9 summarizes the results. The value of Δx is higher using the original vision module for every ball speed, and increases with speed. With Wave3D, the value of Δx is lower and seems to be constant with respect to speed.

5 Conclusion

We presented a novel algorithm, Wave3D, for visual object detection suitable for autonomous robots embedded in complex tasks, which need to save computational power from vision to behavior processing. Furthermore the algorithm allows for the processing of the most updated visual frame leading to the control algorithm to select actions based on the most updated sensory information processed.

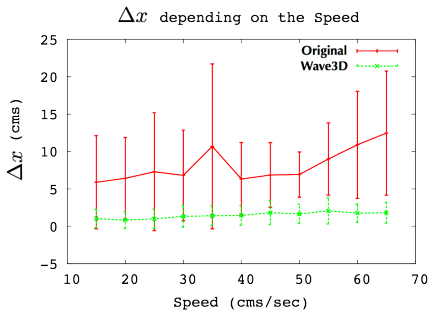

Fig. 9 Mean and standard deviation of Δx using the original vision module (dashed line) and Wave3D (solid line).

The Wave3D algorithm creates and maintains hypotheses of the presence of the objects in the real 3D world relative to the robot position. Such hypotheses are hence independent of a specific 2D frame and therefore can be mapped into every frame, as soon as it is available. The 3D hypotheses are validated in each new 2D image. The search for the object processes starting at the hypotheses and spreads to the rest of the image as wave propagations from the dropped hypotheses. Each frame is then processed as a "continuation" of the previous frames. No new frame is ignored during the visual processing time slots, and robot actuation greatly improves. As we have shown in the experiments, this approach improves the actuation decisions by using the most recent information of the objects and allowing the control to take enough computational effort and to use the most updated visual information.

Looking for real objects in the 3D world is more natural for detecting them. This makes Wave3D more efficient in the object detection process. The execution frequency of the overall robot system is improved as well, without losing perceptual accuracy or reliability, which means more computation available for task-related behaviors. Wave3D is used in the RoboCup domain, but it is applicable to any object detection task for robots which use cameras as main sensor.

References

1. Henry A. Rowley, Shumeet Baluja and Takeo Kanade, *Neural Network-Based Face Detection*, IEEE Transactions On Pattern Analysis and Machine intelligence, Vol. 20, pp. 23–38 (1998)
2. Bernd Heisele, Purdy Ho, Tomaso Poggio, Purdy Ho and Tomaso Poggio, *Face Recognition with Support Vector Machines: Global versus Component-based Approach*, In Proc. 8th International Conference on Computer Vision, pp. 688–694 (2001)
3. Matthias Rtsch, Clemens Blumer, Gerd Teschke and Thomas Vetter, *3D Cascaded condensation tracking for multiple objects*, In Proc. 7th IASTED International Conference Signal Processing, Pattern Recognition and Applications, pp. 361–368 (2010)
4. Simon Denman, Todd Lamb, Clinton B. Fookes, Sridha Sridharan and Vinod Chandran, *Multi-sensor tracking using a scalable condensation filter*, Proceedings of the International Conference on Signal Processing and Communication Systems, (2007)
5. Pratim Ghosh, B.S. Manjunath and K.R. Ramakrishnan, *A compact image signature for RTS-invariant image retrieval*, IEEE International Conference on Visual Information Engineering (VIE 2006), pp. 304–308 (2006)
6. Benjamin Coifman, David Beymer, Philip McLauchlan and Jitendra Malik, *A real-time computer vision system for vehicle tracking and traffic surveillance*, Journal of Transportation Research Part C: Emerging Technologies, Vol. 6, pp 271–281 (1998)
7. Luis M. Fuentes and Sergio A. Velastin, *People tracking in surveillance applications*, Journal of Image and Vision Computing, Vol. 24/11, pp. 1165–1171 (2006)
8. Hastings, W. K., *Monte Carlo sampling methods using Markov chains and their applications*, Biometrika, Vol. 57/1, pp. 97–109 (1970)
9. Frank Dellaert, Steven M. Seitz, Charles E. Thorpe and Sebastian Thrun, *EM, MCMC, and Chain Flipping for Structure from Motion with Unknown Correspondence*, Journal of Machine Learning, Vol. 50/1-2, pp. 45–71 (2003)
10. Hue, C., Le Cadre, J.P. and Perez, P. , *Sequential Monte Carlo methods for multiple target tracking and data fusion*, IEEE Transactions on Signal Processing, Vol 50/2, pp. 309–325, (2002)
11. Pablo Barrera, José M. Caas, Vicente Matellán and Francisco Martín, *Multicamera 3d tracking using particle filter*, Int. Conf. on Multimedia, Image processing and Computer Vision, (2005)
12. Nils J. Nilsson, *Shakey The Robot*, Technical Report AI Center, SRI International (1984)
13. James Bruce, Tucker Balch and Manuela Veloso, *Fast and Inexpensive Color Image Segmentation for Interactive Robots*, In Proceedings of IROS-2000, pp. 2061–2066 (2000)
14. H. Levent Akin, Tekin Mericli, N. Ergin zkucur, Can Kavakhoglu and Bari Gkce, *Cerberus'10 SPL Team*, Technical Report Bogazici University, Department of Computer Engineering (2010)
15. Tim Laue, Thijs Jeffry de Haas, Armin Burchardt, Colin Graf, Thomas Röfer, Alexander Härtl and Andrik Rieskamp, *Efficient and Reliable Sensor Models for Humanoid Soccer Robot Self-Localization*, Proceedings of the Fourth Workshop on Humanoid Soccer Robots in conjunction with the 2009 IEEE-RAS International Conference on Humanoid Robots. pp. 22–29 (2009)