Human behaviour analysis and event recognition at a point of sale

R. Sicre
MIRANE S.A.S.
Cenon, France
sicre@labri.fr

H. Nicolas
LaBRI, University of Bordeaux
Talence, France
nicolas@labri.fr

***Abstract*— This paper presents a new application that aims at improving communication and interactions between digital media and customers at a point of sale.**

Our system analyzes in real-time human behaviour while shopping. In particular, the system detects customer's interest in products and interaction such as people grabbing products.

This system is based on a behaviour model. A video analysis module detects motion, tracks moving object, and describes local motion. Then specific behaviours are recognized and sentences are generated. Finally, our approach is tested on real video sequences.

Keywords: Computer vision; Human behaviour analysis; Video-surveillance; Marketing.

I. INTRODUCTION

Computer vision applications are developed in various fields such as surveillance [11], traffic monitoring [10], video games, marketing, etc.

The field of marketing has evolved lately, with an extensive use of digital media at point of sale. This media offer a new form of communication with customers that brings along new issues. For example, playing advertising clips one after another does not have significant impact on customer. Thus, two major information must be identified: media location and media content. Nowadays a few software systems exist that help solving these challenges. Concerning media location, some systems track customers to obtain statistical information regarding their habits and displacements inside the shopping mall. Other systems calculate directly the media audience, using face detection, to evaluate media impact.

The study, introduced in this paper, is in the same context and aims at adapting the displayed clips to the behaviour of the present customer. We want to improve interaction between customer and media to improve its impact. Our system requires a scene analysis that leads to an interpretation of the customer's behaviour. In particular, we detect people grabbing objects in known areas in real-time. The detection of such an event results in playing a clip related to the specific product, for example.

After a short review, we present the system: first the behaviour model and the video analysis module, then behaviours recognition and semantic interpretation, see figure 1. We show some results and conclude with future work.

II. PREVIOUS WORK

This section presents behaviour analysis and semantic description of behaviour. Our behaviour analysis and semantic interpretation are based on a video analysis module composed of motion detection and object tracking [15]. For a general overview, the reader can refer to several surveys [5] [12] [10].

A. Human behaviour analysis

Human behaviour can be identified in many different contexts [6] [5] [12]. Humans are considered as deformable objects. The goal of behaviour analysis is to recognize motion samples in order to draw high-level conclusion. There are several issues due to the fact that we match real-world activities to outputs perceived by a video processing module. We have to select relevant properties computed with video processing tasks and handle the incompleteness and uncertainty of these properties.

The analysis is usually made in two steps: description and recognition of actions. The first step is to define a model that describes each relevant action in our specific application context. Then there are two main possibilities. First, there is a training phase using labeled data and then data is classified based on the training set, using learning method as Hidden Markov Model, Neural Networks [14], Support Vector Machine (SVM), etc. Secondly a logical model is generated that do not always require a training phase [1] [7]. However these last methods are not very flexible because they rely on scene knowledge.

In our study we combine a logical model, using finite state machine, with a learning method based on SVM.

B. Semantic description of behaviour

Many applications require a description of object behaviour in natural language, suitable for non-specialist operator. There are two main categories of behaviour description methods. Statistical models, like Bayesian network model interpret events and behaviour as interactions between objects by the analysis of time sequences and statistical modeling [14]. Formalized reasoning represents behaviour patterns using symbol systems then recognizes and classifies events with reasoning methods [8]. We choose formalized reasoning for its simplicity.

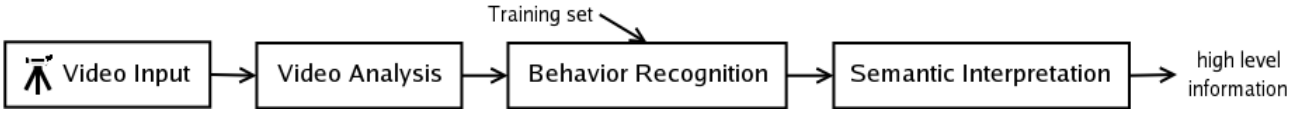

Figure 1. Motion detection on videos from LAB1 (left) and MALL1 (right). Frames are on the first row, rough motion detection on the second, and filtered result on the third.

III. BEHAVIOUR MODEL

This section presents the model that defines human behaviour while shopping. At a point of sale, customers walk around products, look at prices, pick up products, etc. We create six states that correspond to the current behaviour of a person. The chain of states describes the scenario that the person plays.

- **Enter**: A new person appears in the scene.
- **Exit**: The person leaves the scene.
- **Interested**: The person is close to products, i.e. possibly interested.
- **Interacting**: The person interacts with products, is grabbing products.
- **Stand by**: The person is in the scene but not close to any product area or image boundary. The person can be walking or not.
- **Inactive**: The person has left the scene.

IV. VIDEO ANALYSIS

In order to detect the six states, we require information concerning every person in the scene. For every frame, we need people's location, contours, etc. Therefore we use a motion detection and object tracking process. The areas where products heap are located are assumed to be known. During the initialization of the system the user can manually select the products areas. Then, the identification of product grabbing events is done through a description and classification of a person local motion, position relative to product areas, and overlapping surface with product area.

A. Motion detection and object tracking

Motion detection and object tracking identify contours and location of every moving object in the scene for every frame. The method used is divided in two phases: first, motion detection finds moving regions that do not belong to the background. Then, these regions are tracked over the frame sequence. Fast methods were selected in order to cope with the real-time constraints of our final application.

Motion detection uses a pixel based model of the background. A mixture of Gaussians is associated with each pixel, in order to characterize the background [13] [16]. This model is updated on-line. A Gaussian distribution is matched to the current value of each pixel. If this Gaussian belongs to the background, the pixel is classified as such. Otherwise it is considered as foreground, see figure 2. Morphological filters are finally applied on this result.

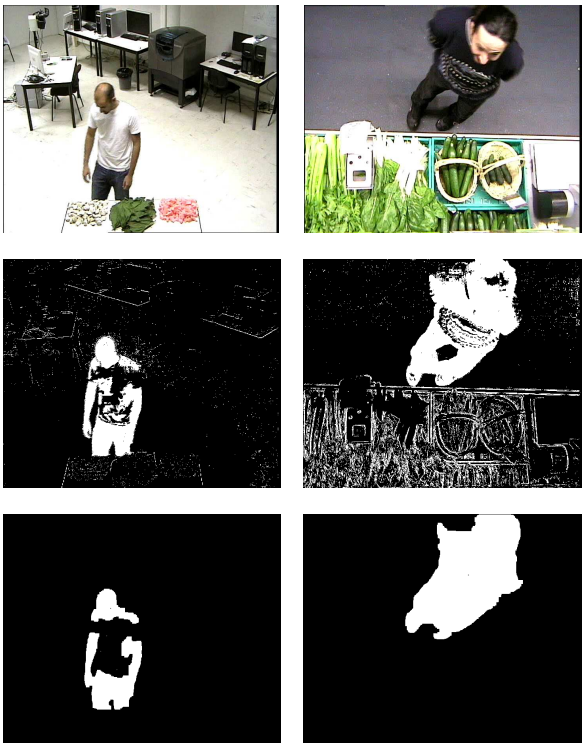

Figure 2. Motion detection on videos from LAB1 (left) and MALL1 (right). Frames are on the first row, rough motion detection on the second, and filtered result on the third.

In practice, a detected object, or person, can be covered by several disconnected regions, because the algorithm misses part of the person, see figure 2. Thus, we need to merge regions. Our merging process is separated in two phases.

The first step checks overlapping regions bounding boxes. Overlapping surface areas are calculated. If these overlapping areas are bigger than a given threshold, regions are merged, as well as if a region is inside another region's bounding boxes.

The second step selects regions close one to another and regions with small overlapping surface areas. These selected regions are listed as potential merge.

We create new regions that are the merge of each couple of selected regions. These new regions are added to the list of newly detected regions. Then the matching process, described in the following paragraph, matches this list of new regions with previously tracked regions. Current regions that are the most similar to previous frame regions are matched and considered as the most relevant.

Object Tracking is composed of two processes. First, we calculate, for each region, a descriptor based on its position, size, surface area, first and second order color moments. These descriptors are then matched from one frame to the next, using a voting process that determines regions that are the most similar in the two sets of regions. Secondly, we use matched regions to build and update the object list. An object is a region that was tracked on several frames. Newly matched regions are used to update information about objects: location, size, colors, etc. Then unmatched regions are compared to inactive unmatched object to solve miss-detections. We also detect regions split and merge that help detecting occlusions.

B. Grabbing product event description

This section focuses on tracked person's interactions with product areas, i.e. people grabbing products. While grabbing a product, a person first reaches out with its arm, then grasps a product, and finally take the product. These different phases in the "product grabbing" event correspond to observable local motion of the person over a local period of time. As in [3] and [14], we define local motion descriptor. However this descriptor is not robust enough to noise and since we do not want to use a long training, we decide to create an interaction descriptor to help characterize product grabbing events. These descriptors are used in the behaviour recognition phase, see Section 5, and are defined as follows.

Local motion descriptor is built for each frame for every tracked person, see figure 3. First, each person's mask is scaled to a standard size of 120x120 pixels, while keeping aspect ratio. Then, the optical flow is computed using Lucas Kanade algorithm [9]. The result of this process is two matrixes with values of motion vectors along x and y axis. We separate negative from positive values in the two matrixes, and end out with 4 matrixes before applying a Gaussian blur on each of them, to reduce the effects of noises. A fifth matrix is computed that represent the person's silhouette. We reduce the dimensionality of these matrixes to save computation time. Each matrix is divided into a 2x2 grid. Each grid cell gets its values integrated over an 18-bin radial histogram (20 degrees per bin). Matrixes are now represented by a 72 (2x2x18) dimensional vector. The full frame descriptor possesses then 360 (72x5) dimensions.

To take into account temporal information, we use 15 frames around the current frame and split them in three sets of 5 frames: past, current, and future. After applying Principal Component Analysis on each set's descriptors, we keep the first 50 components for the current set, while we only keep the first 10 components for the past and future sets. The motion context descriptor possesses then 70 (10+50+10) dimensions and is added to each frame descriptor.

Interaction descriptor uses information coming from the object tracking process. Six values are used: the person's surface covering a product area, a Boolean that rises when this covering surface is bigger than a theoretical hand size, the surface of the person, the height of its bounding box, the position of the top and the bottom of the bounding box along y axis. These measurements, related to the height and

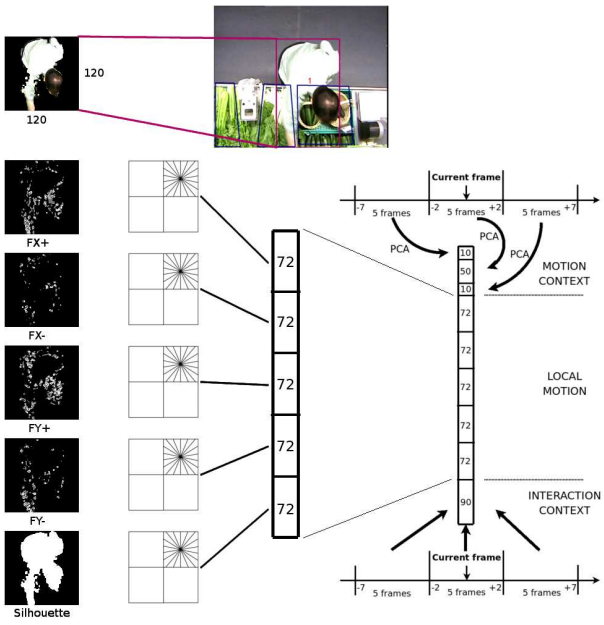

Figure 3. Diagram representing the descriptor.

position of the bounding box, have meaningful variations as a person reaches out for products. We assume that people appears vertically in the scene. The surface tends to increase as a person grasps a product, when products are big enough. These measurements fill the interaction context descriptor that possess 90 (6x15) dimensions, because we keep each measurement of the 15 frames.

V. BEHAVIOUR RECOGNITION

This section presents the behaviour recognition process. Based on the behaviour model, we detect the six states and build a Finite State Machine (FSM).

Using video analysis information, for each frame, the state of each object has to be identified among the six pre-defined states:

Enter is detected when a person appears and is connected to an image boundary.

Exit is detected when a previously tracked person is connected to an image boundary.

Interested is detected when a person's contour connects a product area.

Stand by is detected when a person is in the scene and not connected to a product area or an image boundary.

Inactive is detected when the system loses track of a person. This event happens when a person has left the scene or is occluded by something in the scene, or another person.

Interact is detected using SVM [2] on the local motion and interaction descriptors. First, there are two phases in the data classification: training and testing. The data corresponds to several instances. An instance is composed of a "target value" and several "attributes". In our case, the target value is 0 or 1 depending if a product grabbing event, i.e. Interact state, occurs or not. The attributes correspond to each value of our descriptor. SVM creates a model that predicts target

value from attributes, by solving the following optimization problem.

$$\min_{w,b,\xi} \frac{1}{2} w^T w + C \sum_{i=1}^{l} \xi_i$$

With $y_i(w^T \phi(x_i) + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$

Where y_i are target values, x_i are attributes, ξ_i is the error in the training set, vector w and scalar b are the parameter of the hyperplane. Finally, $C > 0$ is the penalty parameter of the error term.

Training vectors x_i are mapped into a higher dimensional space by the function ϕ. In this higher dimensional space, SVM finds a separating hyperplane that maximize the margin. The system uses a radial basis function (RBF) kernel:

$$K(x_i, x_j) \equiv \phi(x_i)^T \phi(x_j) = \exp(-\gamma \left\| x_i - x_j \right\|^2), \gamma > 0$$

Although, different kernels exist, we choose RBF because it handles nonlinear relations between attributes and target values, unlike linear kernel. RBF also has less hyper parameters and less numerical difficulties than polynomial or sigmoid kernels.

Finite State Machine is used in order to organize and prioritize the six states [1] [7]. The state machine is synchronous and deterministic. Synchronous means that the machine iterates over each new frame. Based on the previous state, the system calculates a new one by testing each transition condition. If a condition is satisfied, the system moves to the new state. Otherwise, the system stays in the same state. The machine is deterministic because for each state, there can not be more than one transition for each possible input. One FSM model the behaviour of one person. We save the person's path through its FSM in order to interpret a high-level scenario. Figure 4 shows a possible path through the FSM.

VI. SEMANTIC INTERPRETATION

This section aims at describing a person's behaviour in natural language and is based on all previous analysis. The generated sentences summarize the current state of action of a person.

Expressing human activities is accomplished using case frames [8] [4]. This tool links cases to sentences in natural language. We use simple case frames composed of three categories as illustrated in the following example:

[AG: "Person 1", PRED: "interacts with", LOC: "area 1"]

AG, PRED and LOC are the agent, the predicate, and the locus of the described action respectively. These three cases allow us to describe all the actions we look for.

Sentences are built using a hierarchy of actions (HoA) that is composed of case frames to represent each node. A parent node derives its children by redefining the verb (predicate is modified) or the locus (locus is modified) as shown in figure 5. In particular, by going through the HoA,

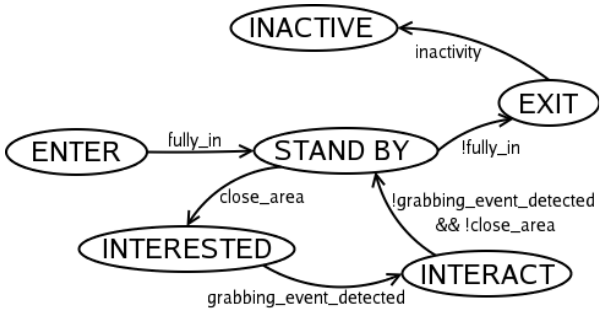

Figure 4. Part of the Finite State Machine, all transitions are not written to make it clear.

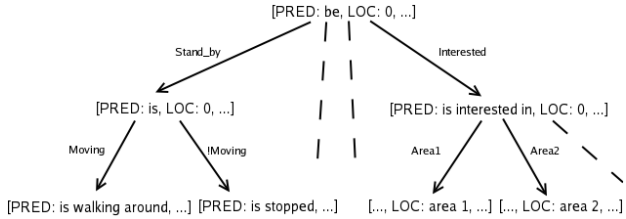

Figure 5. Sample of the hierarchy of action.

the case frame is refined until a final node is reached. Final nodes do not have children and contains all the components of the sentence we want. Here are the final node predicates:

is walking around - is stopped - enters the scene - exits the scene - is interested in - interacts with - is gone

We explain in the algorithm that follows how case frames are generated from the HoA.

1. Let *s* be the current final node. For each new frame, case frames are evaluated from the top of the HoA to determine the new final node *s'*.
2. If the node *s* and the node *s'* are identical, no case frame is generated, and we go back to step 1.
3. Otherwise, a case frame associated to the new state *s'* is generated, and we go back to step 1.

More specifically, when a state change occurs, a case frame is generated, and a sentence is printed.

VII. RESULTS

A. Data description

We use different datasets taken with the same camera. A part of the datasets was taken in our laboratory (LAB1, 2, and 3). The others were taken in a real shopping mall (MALL1 and MALL2). The two first datasets (LAB1 and MALL1) possess five and six sequences respectively and contains a lot of interactions with products. Two and four different people are shopping respectively, see figure 2. LAB2 is a dataset where there is no interaction with the products and has three different actors. Objects or products taken by people have different shapes, colors, and sizes in the video sequences. Also, all products are identical in the heaps. LAB3 and MALL2 are two datasets where multiple

people interact together. Two to four people interact simultaneously, see figure 6.

B. Tests on datasets

We run tests on various video sequences. As the system generates a state for each detected object, for each frame, we compare these results to the ground truth that was labeled by hand. Then, we calculate the percentage of correct states, see table 1. The dataset LAB2 does not contain interaction with products. This dataset has better results than the LAB1, and we conclude that the system is more precise when there are fewer states to detect.

In order to recognize product grabbing events, we use a cross validation process, see table 2. In other words, to recognize events on a video, we use all the other sequences of the dataset as training and then calculate recall and precision. We note that using this process, we only use a couple of minutes of training with a few people.

The dataset LAB1 gives better percentage results than MALL1. This difference is mainly due to noises that are more significant in MALL1 as well as slight camera motion during the capture. However, MALL1 performs very well on recall and precision for the Interact state due to the position of the camera, located directly above the products and closer to them than on LAB1, see table 2. We understand that the camera location is really important to maximize the accuracy of the system. A position close to the products performs better on Interact state recognition. However, having the camera too close to the products make us lose information about the customers, since they are only detected when they are near the products.

After preliminary tests, the local motion descriptor alone offers poor results for the Interact state detection, due to the small training and noises. This is the reason why, we test this descriptor combined with the interaction context descriptor. As we see on table 2, we compare result using only the interaction context descriptor (ICD) and both motion and interaction context descriptor (MID). The MID performs as good as ICD for precision, but offers better results for recall on the first datasets. On multiple people datasets, MID performs slightly better than ICD on average, however local motion description tends to be noisier, due to a few object tracking mismatches. The ICD remains robust in these situations and the recognition rates in multi-person datasets are as good as in the first datasets, see table 2.

The semantic interpretation offers interesting results, see figure 6. Since this interpretation directly relies on the state recognition, a few errors occur. For example, miss-detections occur on the Exit state on MALL datasets, due to the camera position, see figure 6 second sequence.

C. Computation time

Finally, the application has to generate responses quickly as soon as a specific event is detected. The program is tested on a computer with a Pentium 4, 3 Ghz and 1Gb of RAM. The application can analyze 6 to 10 frames per second for an image resolution of 704x576 or 640x480 respectively. Motion detection is the most computational expensive process.

VIII. Conclusion

This paper presents a novel type of application, using computer vision in the field of marketing, to improve interaction between customers and digital media. The system detects, tracks, and analyzes behaviours of shoppers regarding their actions, interests, and interactions with products at a point of sale. The system is tested and offers interesting results, 73% of the frames are correctly labeled, for sequences taken in a real environment. Interactions with products are well detected with a precision of 0.79 and a recall of 0.85. This evaluation helped us understand how the system behaves to maximize its efficiency. A prototype will soon be tested for a long period of time.

As future work, the method can be improved with algorithms that better manage occlusions. It would also be interesting to look for other behaviours and scenarios that can be characterized and detected using this technique.

References

[1] F. Bremond, G. Medioni, “Scenario recognition in airborne video imagery”, *Int. Work. IVM*, pp 57-64. 1998.

[2] C. Chang and C. Lin, “LIBSVM: a library for support vector machines”, http://www.csie.ntu.edu.tw/~cjlin/libsvm, 2001.

[3] A. A. Efros, A.C. Berg, G. Mori, and J. Malik, “Recognizing action at a distance”, *ICCV*, 2003.

[4] C.J. Fillmore, *The case for case*, E. Bach and R. Harms editors, pp 1-88, 1968.

[5] W. Hu, T. Tan, L. Wang, S. Maybank “A survey on visual surveillance of object motion and behaviours,” *IEEE Transac. on syst., man, and Cyb*., pp 334 – 352, 2004.

[6] Y. Hu, L. Cao, F. Lv, S. Yan, Y. Gong, T. Huang, "Action Detection in Complex Scenes with Spatial and Temporal Ambiguities," *Proc. ICCV*, 2009.

[7] N. Ikizler, D. Forsyth, “Searching video for complex activities with finite state models”, *Proc. CVPR*, 2007.

[8] A. Kojima, T. Tamura, K. Fukunaga, “Natural language description of human activities from video images based on concept hierarchy of actions”, *Int. J. Comput. Vis*, vol. 50, n. 2, pp 171 – 184, 2002.

[9] B. D. Lucas and T. Kanade, “An iterative image registration technique with an application to stereo vision”, *in Proc. 7th IJCAI*, pp.674–679, 1981.

[10] B. T. Morris and M. M. Trivedi, “A survey of vision-based trajectory learning and analysis for surveillance,” *Trans. Circ. Syst. Video. Tech*., v. 18, n 8, pp. 1114–1127, 2008.

[11] PETS: Performance Evaluation of Tracking and Surveillance, http://winterpets09.net/

[12] R. Poppe, “A survey on vision-based human action recognition”, *Im. & Vis. Comp. J.*, v. 28, pp. 976-990, 2010.

[13] C. Stauffer, W. Grimson, “Adaptive background mixture models for real-time tracking” *CVPR*, pp. 246–252, 1999.

[14] D. Tran, A. Sorokin, “Human activity recognition with metric learning”, *Euro. Conf. on Computer Vision*, 2008.

[15] A. Yilmaz, O. Javed, M. Shah, “Object tracking: a survey”, *ACM Computing Surveys*, 2006.

[16] Z. Zivkovic, F. van der Heijden “Efficient adaptive density estimation per image pixel for the task of background subtraction” *Pattern Recognition Letters*, vol. 27, n 7, 2006.

TABLE I. TABLE REPRESENTING THE PERCENTAGE OF CORRECTLY DETECTED STATES FOR EACH FRAME, ON VIDEOS OF VARIOUS DATASETS.

Dataset	Video	Frames	Correctness
MALL 1	1	327	70,03%
	2	444	74,77%
	3	434	66,13%
	4	336	70,83%
	5	164	76,22%
	6	232	79,74%
	mean		**72,95%**
LAB 1	1	545	85,87%
	2	672	74,40%
	3	704	76,28%
	4	771	60,57%
	5	518	92,66%
	mean		**76,13%**
LAB 2	1	476	87,61%
	2	342	82,46%
	3	143	91,61%
	4	303	95,71%
	mean		**89,35%**

TABLE II. TABLE REPRESENTING RECALL-PRECISION OF INTERACT STATE DETECTION. TWO DESCRIPTORS ARE TESTED ON VARIOUS VIDEOS: INTERACTION CONTEXT IC AND MOTION AND INTERACTION CONTEXT MI.

Dataset	Video	Frames	Recall IC	Precision IC	Recall MI	Precision MI
MALL1	1	327	0,5306	0,5977	0,8163	0,6667
	2	444	0,9307	0,9592	0,9505	1
	3	434	0,6667	0,7368	0,7525	0,5802
	4	336	0,6905	0,9063	0,7976	0,8701
	5	164	0,5	1	0,5	1
	6	232	0,8475	0,9434	0,8983	0,9815
	mean		**0,6943**	**0,8572**	**0,7859**	**0,8498**
LAB1	1	545	0,7299	0,7692	0,8321	0,8085
	2	672	0,6585	0,648	0,6748	0,6288
	3	704	0,6774	0,9333	0,7473	0,9392
	4	771	0,7203	0,7687	0,7552	0,7347
	5	518	0,9818	0,75	0,9818	0,7941
	mean		**0,7536**	**0,7738**	**0,7982**	**0,7811**
MALL2 MP	1	215	0,8929	0,8333	0,8214	0,902
	2	208	0,6462	0,8235	0,7846	0,8947
	3	735	0,7151	0,7278	0,9101	0,72
	4	153	1	0,5106	0,5417	0,9286
	5	382	0,8333	0,8404	0,6583	0,8404
	6	504	0,7273	0,8571	0,8561	0,8828
	mean		**0,8025**	**0,7655**	**0,762**	**0,8614**
LAB3 MP	1	212	0,7282	0,8929	0,8738	0,9375
	2	211	0,9063	0,8969	0,7604	0,9012
	3	300	0,784	0,7538	0,856	0,7643
	4	303	0,7561	0,5636	0,7317	0,6383
	5	259	0,8158	0,6596	0,8026	0,6854
	mean		**0,7981**	**0,7534**	**0,8049**	**0,7853**

Figure 6. Results for MALL 2 video 1

Figure 7. Results for LAB 1 video 1