Object Tracking Implementation for a Robot-Assisted Autism Diagnostic Imitation Task

Kruno Hrvatinić, Luka Malovan, Frano Petric, Damjan Miklić and Zdenko Kovačić
University of Zagreb, Faculty of Electrical Engineering and Computing, LARICS Laboratory, Unska 3, 10000 Zagreb, Croatia
Email: larics@fer.hr

***Abstract*—Autism spectrum disorders (ASD) is a term used to describe a range of neurodevelopmental disorders affecting about 1% of the population, with increasing prevalence. Due to the absence of any physiological markers, diagnostics is based purely on behavioral tests. The diagnostic procedure can in some cases take years to complete, and the outcome depends greatly on the expertise and experience of the clinician. The predictable and consistent behavior and rapidly increasing sensing capabilities of robotic devices have the potential to contribute to a faster and more objective diagnostic procedure. However, significant scientific and technological breakthroughs are needed, particularly in the field of robotic perception, before robots can become useful tools for diagnosing autism. In this paper, we present computer vision algorithms for performing gesture imitation. This is a standardized diagnostic task, usually performed by clinicians, that was implemented on a small-scale humanoid robot. We describe the algorithms used to perform object recognition, grasping, object tracking and gesture evaluation in a clinical setting. We present an analysis of the algorithms in terms of reliability and performance and describe the first clinical trials.**

I. Introduction

Autism spectrum disorder (ASD) is a developmental disorder characterised by impairment in social interaction, verbal and nonverbal communication and by repetitive behaviours and interests. It has become a commonly diagnosed neurodevelopmental disorder, with increasing prevalence rates, affecting about one in every 100 children [8], and there are no medical markers of autism that could be used in a diagnostic process. Therefore, the diagnosis of ASD is based solely on behavioural observations made by experienced clinicians. However, specific behaviors that are included in diagnostic frameworks and the point at which individual differences in behavior constitute clinically relevant abnormalities are largely arbitrary decisions [4]. *There exists a need for quantitative, objective measurements of social functioning for diagnosis, for evaluating intervention methods, and for tracking the progress of individuals over time* [6]. Modern robotic devices have the potential to fill this need. They can perform actions consistently and repeatably, and evaluate the child's reactions in a quantitative and unbiased way. The tendency of autistic children to interact with technical devices more than with humans around them is another argument in favor of employing robotics for improving the autism diagnostics procedures.

As a starting point for our work, we have chosen the ADOS test [3], which represents the state of the art in autism diagnostics. As described in a preliminary report [5], we have chosen four tasks from the ADOS test and adapted them to the capabilities of the Nao [1] humanoid robot. All of the tasks require the robot to perform a specific action and monitor and evaluate the reaction of the child. In this paper, we present the computer vision algorithms that we have implemented in order to perform the *Functional and symbolic imitation task.* We describe the algorithms and present preliminary results of evaluating their effectiveness in a clinical setting.

The paper is organized as follows. In Section II, we describe the imitation task in detail, and mention the specific circumstances relevant for image processing algorithm design. The three major components of our processing pipeline, namely image segmentation, object tracking and gesture recognition, are described in Sections III, IV, V respectively. Algorithm evaluation results are described in Section VI, and concluding remarks are given in Section VII.

II. Imitation task description

The purpose of the imitation task is to assess the ability of a child to perceive and reproduce a demonstrated gesture. From the point of view of a robotic examiner, the task has three phases:

1) Demonstration,
2) Encouragement for imitation,
3) Observation.

To start the task, the robot needs to make sure the child has its head turned towards it. This can be accomplished by using the robot's built-in face detection software. In the demonstration phase, the robot picks up an object, for example a cup, and demonstrates a gesture, such as drinking, accompanied by appropriate sounds. It then places the object in front of the child and encourages it, by speaking and pointing, to perform the same gesture. Finally, it tracks the object in order to estimate and evaluate the child's reaction. The task can be performed using an object with an obvious everyday function, e.g., using a cup to demonstrate drinking, which is called *functional imitation*. It can also be performed using for example a wooden cylinder to demonstrate the flight of an aeroplane, in which case it is called *symbolic imitation*.

The hardware platform used in our research is the Nao H25 v4.0 humanoid robot, equipped with a 960p30fps RGB camera. In terms of image processing algorithms necessary for successfully conducting the imitation task, the robot needs to perform image segmentation, in order to find the manipulated object. After demonstrating the gesture and releasing the object, the robot needs to keep tracking the object in order to evaluate the gesture reproduced by the child. Since we are considering autism diagnostic procedures performed in clinical settings, we have some control of the environment layout, and the number and type of objects that are visible to the robot.

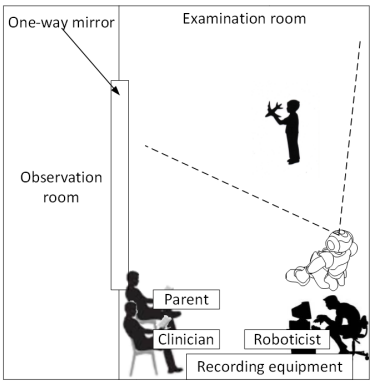


Fig. 1: Layout of the examination room.

Taking into account the layout of the examination room that was used for our research, shown in Figure 1, we have made the following simplifying assumptions:

1) the object can be segmented by color, i.e., the color of the manipulated object is significantly different from the background),
2) the only person in the area of the room visible to the robot is the child, i.e., if the object moves, it is because either the robot or the child are moving it.

On the other hand, the following requirements must be satisfied by the image processing algorithms:

1) the background is dynamic because of robot motion,
2) multiple objects need to be tracked simultaneously, in order to register both the motion of the imitation object and the motion of the child,
3) objects need to be tracked during temporary occlusions,
4) the algorithms need to provide soft real-time performance, running on the robot's onboard computer.

III. Image segmentation

Image segmentation is the first step of the image processing pipeline. It separates the objects of interest, which are in this case the imitation object and the child's hands and face, from the rest of the image. We have extended the work described in [2], based on histogram back projection, to allow simultaneous segmentation of several objects. We make use of the assumption that the tracked object is of a specific, uniform color, which is not over represented in the environment.

A. Histogram creation

Based on [7], [9], the YUV colorspace has been chosen for segmentation. For each class of tracked object, an U-V histogram must be provided (Y data is discarded), constructed from previously gathered training data. If we denote by $N_o(k), k \in [1, K] \subset \mathbb{N}$ the number of pixels belonging to an object, that are grouped in histogram bin k (out of a total of K bins)[1], then the probability distribution of a given color interval can be approximated by

$$P(k|o) \approx \bar{N}_o(k) = \frac{N_o(k)}{\sum_{i=1}^{K} N_o(i)}. \quad (1)$$

[1]The implemented algorithm uses $64 \times 64 = 4960$ bins. The number of bins per dimension is four times smaller than the data resolution to provide some robustness to slight color variations

Similarly, the a priori probability of a color interval occurring can be approximated by

$$P(k) \approx \bar{N}_{bg}(k) = \frac{N_{bg}(k)}{\sum_{i=1}^{K} N_{bg}(i)}, \quad (2)$$

where $N_{bg}(k)$ is the total number of pixels in the image that are sorted in interval k. Finally, the a priori probability of a pixel belonging to the object can be approximated by

$$P(o) \approx \frac{\sum_{i=1}^{K} N_o(i)}{\sum_{i=1}^{K} N_{bg}(i)}. \quad (3)$$

Combining (1), (2) and (3) by the Bayes rule

$$P(o|k) = \frac{P(k|o)P(o)}{P(k)}, \quad (4)$$

we obtain (after canceling corresponding terms)

$$P(o|k) \approx \frac{N_o(k)}{N_{bg}(k)}, \quad (5)$$

which says that the probability of a pixel of some known color being part of an object can be approximated by dividing the object histogram with the image histogram. That probability will always be less then or equal to 1 because $N_o(k) \leq N_{bg}(k)$ and it will be equal to 1 for object colors only. Several pictures are used, obtained in different lighting conditions and against different backgrounds, to create a histogram for each object with background. For each image, a second histogram is created by including only manually selected object pixels, and then apply equation (5) on this pair of histograms. This process allows for a segmentation procedure that is more robust to varying lighting conditions and the presence of similar colors in the background.

B. Image binarization

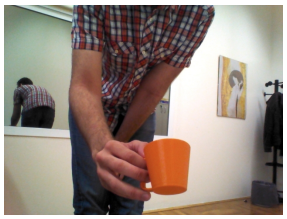
In order to identify objects in the input images obtained by the robot's camera, the image is first converted to YUV space, and then filtered with a small window size median filter to reduce noise. For every object we wish to track, *histogram back projection* is performed on the input image. This yields one single channel image of probability per object. Separating the object from the background is done by *hysteresis thresholding*. In the first step of this procedure the image is binarized with a fixed higher threshold. This yields a sub segmented image: parts that surely belong to the object are separated, but some parts of the object are missing. The area of the object is then expanded by assigning to it adjacent pixels whose value is greater than the lower threshold. This procedure is capable of accurately segmenting an object while ignoring areas of similar color that belong to the background.

Examples of the hysteresis thresholding procedure are shown in Figure 2. Darker green areas in Figure 2b represent areas segmented with the higher threshold, lighter green areas were selected by the lower threshold. The bottom figures demonstrate the ability of the algorithm to reject background artifacts similar in color to the tracked object. The areas colored red in Figure 2d, which satisfy only the lower threshold, are rejected as outliers.

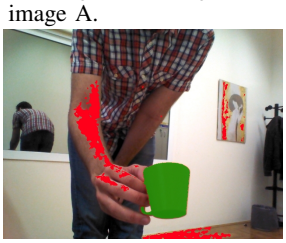
As noted earlier, each object of interest is segmented individually, yielding one binary image per object. All the

(a) Original image A.

(b) Segmented regions of image A.



(c) Original image B.



(d) Segmented regions of image B.

Fig. 2: Two examples of the hysteresis threshold procedure.

images are then merged into one scene representation using the binary OR operator. This composite binary image consists of spatially 8-connected groups of pixels which represent one or several objects. Individual objects are then extracted and tracked using combined metrics of distance and color, as described in the following section.

IV. Object tracking

Object segmentation and tracking are complementary procedures in our implementation. Information about object position in the previous time step provides a prediction for its position in the current image. Segmentation provides the update step for the current object position.

A. Object model

For successful tracking, the object needs to be represented by a simple model that captures all relevant information. Because knowledge of the exact shape of the object is not necessary for the tracking task, we use an ellipse model described by $m_i = (c_{xi}, c_{yi}, a_i, b_i, \theta_i)$, where (c_{xi}, c_{yi}) is the ellipse centroid, a_i and b_i are semi-axis lengths and θ_i is the rotation angle relative to the horizontal axis.

The ellipse model is constructed from the group of 8-connected pixels representing the object by calculating the covariance and centroid of their distribution. Let the object be represented by M pixels $\mathbf{p_j} = [x_j, y_j]^T, j \in [1, M] \subset \mathbb{N}$. The pixel coordinate distribution is expressed as:

$$E[\mathbf{p}] = \frac{\sum_{j=1}^{M} \mathbf{p_j}}{M} = [\mu_x, \mu_y]^T, \tag{6}$$

$$\mathbf{\Sigma} = \begin{bmatrix} m_{xx} & m_{xy} \\ m_{xy} & m_{yy} \end{bmatrix}, \tag{7}$$

Fig. 3: Examples of ellipse models.

where

$$\mu_x = \frac{\sum_{j=1}^{M} x_j}{M}, \mu_y = \frac{\sum_{j=1}^{M} y_j}{M}, \tag{8}$$

$$m_{xx} = \frac{\sum_{j=1}^{M} (x_j - \mu_x)^2}{M}, m_{yy} = \frac{\sum_{j=1}^{M} (y_j - \mu_y)^2}{M}, \tag{9}$$

$$m_{xy} = \frac{\sum_{j=1}^{M} (x_j - \mu_x)(y_j - \mu_y)}{M}. \tag{10}$$

Assuming a Gaussian distribution of points with expectation $E[\mathbf{p}]$ and covariance matrix Σ, the semi-axes lengths of an ellipse containing the points within a 90% confidence interval can be obtained as:

$$a = 2\sqrt{s\lambda_1}, b = 2\sqrt{s\lambda_2} \tag{11}$$

where λ_1, λ_2 are the covariance matrix eigenvalues, and s is the value of the χ^2 distribution for two degrees of freedom with $\alpha = 1 - 0.9 = 0.1$. For the selected confidence interval, $s = 4.605$, eigenvalues are calculated by the equation:

$$K = \sqrt{m_{xx}^2 + m_{yy}^2} - 4(m_{xx}m_{yy} - m_{xy}^2), \tag{12}$$

$$\lambda_1 = \frac{m_{xx} + m_{yy} + K}{2}, \lambda_2 = \frac{m_{xx} + m_{yy} - K}{2}, \tag{13}$$

and the tilt angle of the ellipse θ is calculated as:

$$\theta = \arctan\left(\frac{m_{xx} - \lambda_1}{-m_{xy}}\right). \tag{14}$$

Examples of segmented objects and their corresponding ellipse models are shown in figure 3.

B. Assigning pixels to objects

Assigning segmented pixels to each object model combines object model information available from the previous time step (prediction), with the groups of 8-connected pixels obtained by segmentation (correction). We make the following assumptions in the assignment procedure:

1) Each pixel of each group belongs to at least one object
2) A pixel can belong to several objects
3) Only one group of pixels can be assigned to each object

These assumptions take into account cases in which objects are in contact or are covering each other and are met in most real situations. It should be noted that the aforementioned assumptions allow for a situation where no pixels are assigned to an object. This is interpreted as the object having left the scene.

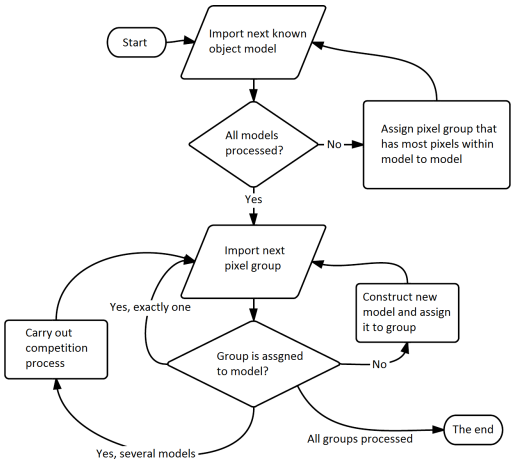


Fig. 4: Flow chart of the procedure for establishing object-pixel correspondence.

In order to perform the assignment, we use the metric

$$d(m_i, \mathbf{p_j}) = \left|\left| \begin{bmatrix} \frac{1}{a_i} & 0 \\ 0 & \frac{1}{b_i} \end{bmatrix} \begin{bmatrix} \cos(\theta_i) & -\sin(\theta_i) \\ \sin(\theta_i) & \cos(\theta_i) \end{bmatrix} \left(\mathbf{p_j} - \begin{bmatrix} c_x \\ c_y \end{bmatrix} \right) \right|\right| \tag{15}$$

which represents the distance of pixel j from object i as the distance of the pixel from the origin of the coordinate system obtained by mapping the ellipse to a unit circle. This metric has the useful property that $d(m_i, \mathbf{p_j}) < 1$ for all pixels inside the ellipse. The assignment procedure is outlined in Figure 4. In the first step, groups of pixels are assigned to objects by calculating the distance metric for each object-pixel pair and, for each object, storing the number of group pixels for which the distance metric is $d(m_i, \mathbf{p_j}) < 1$. This results in a matrix containing the number of pixels which belong to a specific group and also fall within a specific object's ellipse model. After all pixels have been checked, the group which has the largest number of pixels inside an object's ellipse model is assigned to that object. This way at most one group is assigned to each model. If a pixel group was not assigned to any object during this step, it is considered to be a new object.

The second step solves cases in which one group is assigned to several objects, meaning that (partial) occlusion has occurred. To resolve this, those objects must divide the group's pixels among themselves. All the pixels in such contested groups are assigned to one of the contesting objects according to the minimum value of the metric

$$d_2 = d(m_i, \mathbf{p_j})(1 + P_c(\mathbf{p_j})), \tag{16}$$

where $P_c(\mathbf{p_j})$ represents the probability that pixel $\mathbf{p_j}$ belongs to the histogram with which object i is segmented, calculated by taking the histogram value for that pixel's color.

After assigning all segmented pixels to objects (either existing ones or new ones), we perform the prediction step. Assuming that an object in motion will continue its motion in similar direction and with similar speed, the ellipse model center $m_j[k]$ is translated by

$$\overrightarrow{v_2} = 0.5([c_x[k], c_y[k]] - [c_x[k-1], c_y[k-1]])^T \tag{17}$$

The factor 0.5 is introduced because in actual situations such as putting a cup back on the table or clapping hands, sudden deceleration happens more often than sudden acceleration.

V. Gesture recognition

The implemented gesture recognition algorithm offers a simple way of locating the parts of an object's trajectory which match a certain shape. A gesture model is defined as a list of motion directions and compared to the trajectory. This algorithm can determine if a gesture exists in objects trajectory, but can also determine how many times, when and for how long the gesture has been performed.

A. Selection of storage space for trajectories

Assuming the object model is known in every time step, it is possible to construct a trajectory of the object in image space by storing the centroid in every time step. Such a representation is appropriate for use with a fixed camera but fails to accurately describe the object's movement when the camera is in motion, as is the case with a camera mounted to the actuated head of a robot.

Instead, using a properly calibrated pinhole camera model, a line can be drawn through the camera's focus and the recorded point in the image plane. If the pitch and yaw angles α_y, α_z of this line with regards to a coordinate system fixed to the robot's body are used as trajectory points, the trajectory becomes independent of the robot's head rotation. This is the approach used for trajectory storage in the implemented algorithm, as it produces a larger virtual field of view and allows the robot to be placed much closer to the child.

B. Gesture model

In order for a gesture to be detected, it must be described by a model. The model used was selected based on the following requirements:

1) Able to represent gestures in (α_y, α_z) space
2) Invariant to scaling by width
3) Invariant to scaling by height
4) Invariant to translation
5) Not invariant to rotation

The selected model is based on the trajectory angle β, the angle of the object's velocity vector computed as a difference between consecutive trajectory points. Let the beginning of the gesture be defined as an interval χ_1 of angle β i.e. the object beginning to move in a certain direction. Building off this definition, one can define all other segments of the trajectory also with respect to the trajectory angle, which results in the gesture being represented as a set of time-sequential angle intervals. A trajectory will satisfy such a model if its direction smoothly crosses from one interval to another.

To enable modularity and simplify defining new gestures, all gestures are to be described through the use of standard angle intervals which are defined as follows:

$$S(i, \delta) = \left[\frac{2(i-1)\pi}{8} - \delta, \frac{2(i+1)\pi}{8} + \delta \right], \{i \in \mathbb{N} | 0 \leq i \leq 7\} \tag{18}$$

Equation (18) defines eight intervals of width $\frac{\pi}{4} + 2\delta$ with centers in $\frac{i\pi}{4}$. These intervals can be interpreted as the directions $\{right, up-right, up, up-left, left, down-left, down, down-right\}$ with an overlap between adjacent directions of δ.

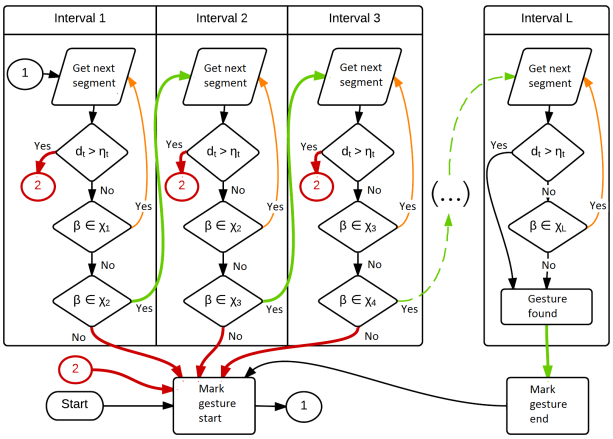


Fig. 5: Flow chart of procedure for comparing model and trajectory

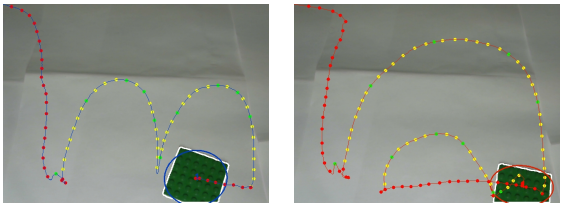


Fig. 6: Comparison procedure

This model meets all the specified requirements and allows for a comparison algorithm with $O(N)$ complexity. It is human-readable and can be made robust to measurement errors and the inadvertent inaccuracy of a person performing the gesture by adjusting the parameter δ of the standard intervals.

C. Comparing models and trajectories

Figure 5 shows the algorithm used for comparing a model with a trajectory. The objective is to find every interval of point indices $[k_{start}, k_{stop}]$ among trajectory points $\Gamma[k] = [t[k], \alpha_y[k], \alpha_z[k]]^T$ in which all the trajectory points represent a gesture defined by the model $\chi = [\chi_1, \chi_2 \cdots \chi_L]$ consisting of L time-sequential angle intervals. The sensitivity of the algorithm can be adjusted by using the distance and time limit parameters η_d, η_t whose default values are $\eta_d = 0.05rad, \eta_t = 1500ms$.

Since the procedure relies on dividing the trajectory into segments, point distances in angle and time need to be defined:

$$d_\theta(\Gamma[s_m], \Gamma[s_n]) = \sqrt{(\alpha_y[s_m] - \alpha_y[s_n])^2 + (\alpha_z[s_m] - \alpha_z[s_n])^2} \quad (19)$$

$$d_t(\Gamma[s_m], \Gamma[s_n]) = t[s_m] - t[s_n] \quad (20)$$

A trajectory segment is defined as an interval of trajectory indices $[s_n, s_m], n > m$ for which $d_\theta > \eta_d$ and/or $d_t > \eta_t$ holds. The next segment of the trajectory can be retrieved by increasing n from $n = m + 1$ until one of these conditions is met. If the stopping condition is distance travelled, the object is moving and the trajectory angle β for the current trajectory segment can be calculated:

$$\beta = \arctan\left(\frac{\alpha_z[s_m] - \alpha_z[s_n]}{\alpha_y[s_m] - \alpha_y[s_n]}\right) \quad (21)$$

The angle β is compared to the angle interval that corresponds to the current state. If β is within the current state's angle interval, the gesture is still being performed and the next segment is retrieved. If β is outside the current state's angle interval, the trajectory has either transitioned into the next state's interval or the part of the trajectory from this part onward does not represent the gesture. If the next state exists and β belongs to its angle interval, the algorithm sets that state to be the current state, otherwise the current state is set to the beginning state and the gesture start point is reset.

If a segment that does not match the current state's angle interval is found, it is possible the gesture has just been successfully completed. This is detected by using a simple rule: if the algorithm is in the last state in the model, the gesture is valid and finished.

Figure 6 shows the algorithm used for comparing the model with the trajectory. Points marked with circles represent the ends of segments and the color of the circle corresponds to the action taken by the algorithm. Green circles represent transitions to the next state or the ending points of the gesture, yellow circles represent segments whose angles β correspond to the current state's angle interval and red circles represent the ending points of segments that do not represent the gesture. The colours on figure 5 also roughly correspond to these transitions.

VI. Experimental evaluation

Gestures that the robot recognizes during the imitation task and their representations are summarized in table I.

TABLE I: Gesture representation

Gesture	Representation	Alternative
Frog	up-left, left, down-left	up-right, right, down-right
Drink	up, down	N/A
Airplane	left, right	N/A

A "frog jumping" gesture is defined as a series of arches (in both directions), with motion having a noticeable horizontal displacement. A "drinking" gesture can be seen as a narrow frog jump, with little or no horizontal displacement, resulting in the gesture being modelled as upward motion followed by downward motion. Due to ambiguity of human interpretation, airplane gestures are simply modelled to be characterized by alternating left and right motion, which may not be the most accurate model but is chosen because there are no parts of the model that are similar to the previous two gestures.

A. Benchmarking

Benchmarking of the segmentation was performed through 1000 images with a framerate of 15 FPS with head tracking turned on and with head tracking turned off. The object histogram used was constructed offline using 40 images of the object in 640×480 resolution. The images were captured using the robot's internal camera under various lighting conditions and backgrounds and the pixels belonging to the object were marked by a human operator using image editing software. The background image was then combined with pixel data to produce a 64×64 bin two-dimensional U-V histogram, as described in Section III. The object was tracked successfully

TABLE II: Benchmarking results

Gesture	Tracking	Tests	As frog	As drink	Success rate
Frog	Head OFF	10	7/10	2/10	7/10
Frog	Head ON	10	5/10	3/10	5/10
Drink	Head OFF	10	0/10	10/10	10/10
Drink	Head ON	10	0/10	10/10	10/10

TABLE III: Comparison of **H**uman and **R**obot observations

Session	Drink		Frog		Airplane	
	H	R	H	R	H	R
Child #1	✓	✓	×	×	✓	×
Child #2	✓	✓	✓	✓	-	
Child #3	Not performed (child not interested)					
Child #4	Not performed (robot malfunction)					

in every image without interruptions, confirming the efficiency of the segmentation algorithm.

Gesture recognition tests were then performed for the frog and drinking gesture, both shown to the robot 10 times.

As table II shows, the drinking gesture was recognized much more consistently than the frog gesture, also, the drinking gesture was not mistaken for the frog gesture while the opposite occurred several times. This is caused by the similarities between these two gestures and variability in human performance. Due to that variability, smaller horizontal displacement during the frog-like gesture can occur, which is in some cases interpreted as upward motion instead of up-right or up-left, resulting in incorrect interpretation of the gestures.

However, in actual use cases the robot is expecting a certain gesture meaning misclassifications are less of an issue. Additionally the frog gesture is expected to be performed by the child multiple times as opposed to the drink gesture which the child performs only once. The algorithm is therefore accurate enough for the purpose it was developed for.

B. Clinical evaluation of imitation task

During the first clinical tests, sessions with four children (three with ASD, one typically developing) were performed, the results of which are presented in table III. Observations made by the robot are compared to those obtained by the expert clinicians, which were additionally validated through analysis of video recordings.

As can be seen in table III, only two sessions were successfully performed, with one failing due to the child not being interested in interaction with the robot, while the other failed due to robot malfunction. In the first session with the child, the robot correctly interpreted the drinking behaviour, while failing to detect the airplane motion. Robot successfully detected that the child did not perform a frog-like gesture. During the session with the second child, the robot correctly detected both the drinking and frog jumping gestures. Since the robot failed to correctly demonstrate the airplane gesture, the results for that part of the task were not obtained.

VII. Conclusions

In this paper, we have described the implementation of an object tracking algorithm for gesture evaluation in the context of robot-assisted autism diagnostics. The purpose of the algorithm is to detect and accurately classify a child's ability to reproduce a simple gesture, such as drinking or the imitation of a jumping frog. The implementation exploits specific conditions of the clinical setting in which the diagnostic procedure is performed. Because the environment is semi-controlled, and the properties of the manipulated objects can be more or less freely chosen, object detection is primarily based on color. Significant features of the described tracking procedure are the ability to track multiple objects simultaneously as well as robustness to background noise and partial object occlusion. It is capable of extracting individual objects and keeping them separated under strong interactions, even when the objects belong to the same class, i.e., are of the same color. The gesture recognition part is based on comparing the observed trajectory to a parametric trajectory description that is compact and invariant to scaling, translation and rotation. Initial deployment in a clinical setting confirmed the usability of the approach.

As the presented work is part of a larger effort aimed at developing a robot capable of acting as an assistant in autism diagnostic procedures, future work will be focused on performing clinical evaluations on a larger population of children. Some drawbacks of the described tracking procedure and classification procedure, such as sensitivity to lighting conditions and handling of full occlusions will be improved.

Acknowledgment

This work has been supported in part by ACROSS - Centre of Research Excellence for Advanced Cooperative Systems (European FP-7 Capacities "Research Potential" program, grant no. 285939, FP7-REGPOT-2011-1) and by the Croatian Science Foundation under the ADORE project (HRZZ-93743-2014).

References

[1] Aldebaran Robotics. *Nao Software Documentation*, v1.14.5 edition, 2013. https://community.aldebaran-robotics.com/doc.

[2] Antonis A Argyros and Manolis IA Lourakis. Real-time tracking of multiple skin-colored objects with a possibly moving camera. In *Computer Vision-ECCV 2004*, pages 368–379. Springer, 2004.

[3] C. Lord, M. Rutter, P.C. Dilavore, and Risi. S. *Autism Diagnostic Observation Schedule*. Western Psychological Services, 2002.

[4] C.F. Norbury and A. Sparks. Difference or disorder? Cultural issues in understanding neurodevelopmental disorders. *Developmental Psychology*, 49(1):45–58, 2013.

[5] F. Petric, M. Cepanec, J. Stošić, S. Šimleša, K. Hrvatinić, A. Babić, L. Malovan, D. Miklić, and Z. Kovačić. Four tasks of a robot-assisted autism spectrum disorder diagnostic protocol: First clinical tests. In *Global Humanitarian Technology Conference (GHTC)*. IEEE, 2014. (To appear).

[6] B. Scassellati, Henny A., and M. Mataric. Robots for use in autism research. *Annual Review of Biomedical Engineering*, 14(1):275–294, 2012.

[7] J-C Terrillon, Mahdad N Shirazi, Hideo Fukamachi, and Shigeru Akamatsu. Comparative performance of different skin chrominance models and chrominance spaces for the automatic detection of human faces in color images. In *Automatic Face and Gesture Recognition, 2000. Proceedings. Fourth IEEE International Conference on*, pages 54–61. IEEE, 2000.

[8] K. Williams, S. MacDermott, G. Ridley, E.J. Glasson, and J.A. Wray. The prevalence of autism in Australia. Can it be established for existing data? *Journal of Paediatrics and Child Health*, 44(9):504–510, 2008.

[9] Ming-Hsuan Yang, David Kriegman, and Narendra Ahuja. Detecting faces in images: A survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(1):34–58, 2002.