Robotic Open-architecture Technology for
Cognition, Understanding and Behavior

Cogsys
Cognitive Systems

Project No. 004370

RobotCub
Development of a Cognitive Humonoid Cub

Instrument: Integrated Project
Thematic Priority: Cognitive Systems

D 3.5 Robotic Implementation of models of sensory-motor coordination for reaching tasks

Due date:
Submission Date:

Start date of Project: **01/09/2004** Duration: 60 months

Organization name of lead contractor for this deliverable: **SSSA**

Responsible Persons: **Cecilia Laschi, Paolo Dario**
Authors: **Lord Kenneth Pinpin, Cecilia Laschi**

Revision: **1.0**

Project co-funded by the European Commission within the Sixth Framework Program (2002-2006)		
Dissemination Level		
PU	Public	
PP	Restricted to other program participants (including the Commission Service)	
RE	Restricted to a group specified by the consortium (including the Commission Service)	
CO	Confidential, only for members of the consortium (including the Commission Service)	CO

Contents

1 Introduction

This deliverable reports the activities and results related to Task 5.6 of WP5 – Imitation. This task was planned as an experimental investigation on the role of gaze in imitation of hand movements. It aimed at studying the role of gaze in the observation of hand movements and at a robotic implementation of an imitative behaviour based on the person's gaze shifts.

This work is based on the neuroscientific findings on the gaze behaviour during manipulation tasks. Land et al. [1] reported that during everyday activities, gaze fixations always are close to the object being manipulated, and very few fixations are irrelevant to the task occurred. Johansson et al. [2] demonstrated that gaze in manipulation tasks consistently fixates future object contact points well before the hand reaches these locations and anticipates reaching trajectory via-points. Flanagan and Johansson further reported in [3] that an observer's gaze behavior while watching a person (actor) performing a task reveals the actor's intention by making predictive eye movements similar to when the observer performs the task himself.

The observation of a person's gaze shifts is used in this work to implement in a robot the capability for predicting and executing the person's hand movements.

2 Proposed approach

The approach proposed in this work is based on two key observations on human gaze behavior.

First, as Johansson reported in [2], a person's gaze fixations marked spatial goals of successive action phases while performing well-learned object manipulation tasks. Gaze and hand movements were linked concerning landmarks with gaze leading. Gaze fixations in landmarks where contact is about to be made were found to be obligatory while those in possible contact points such as obstacles were optional. These gaze fixation areas reveal 'visual control points' for goal completion of each action phase. This suggests that monitoring gaze fixation behavior can reveal task segmentation and is useful for hand motion prediction in the context of a task.

Flanagan and Johansson further observed in [3] that an observer's gaze behavior while watching a person (actor) performing a task reveals the actor's intention by making predictive eye movements similar to when the observer performs the task himself. This behavior suggests that existing sensorimotor representations of familiar tasks are implemented in the observer's brain in real-time (see Fig. 1).

Our idea is to replace the role of the human observer with a robot that tracks the gaze behavior of the actor and uses the observed gaze (instead of the hand movements) to select which among stored action programs to execute (see Fig. 2). Since the observed actor's gaze behavior consists of predictive eye movements that occur in advance of hand movements, detecting these in the form of gaze fixations at obligatory landmark areas (near object contact points) would provide information on what hand movement is about to occur.

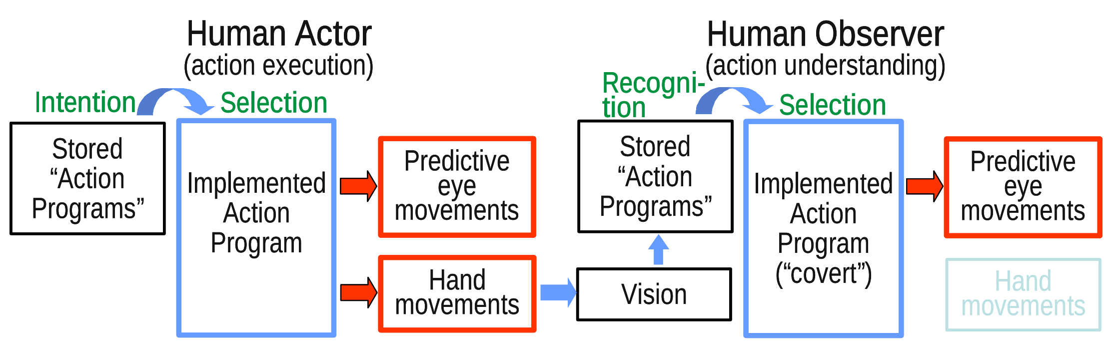

Fig. 1. A human observer's gaze behavior during action observation.

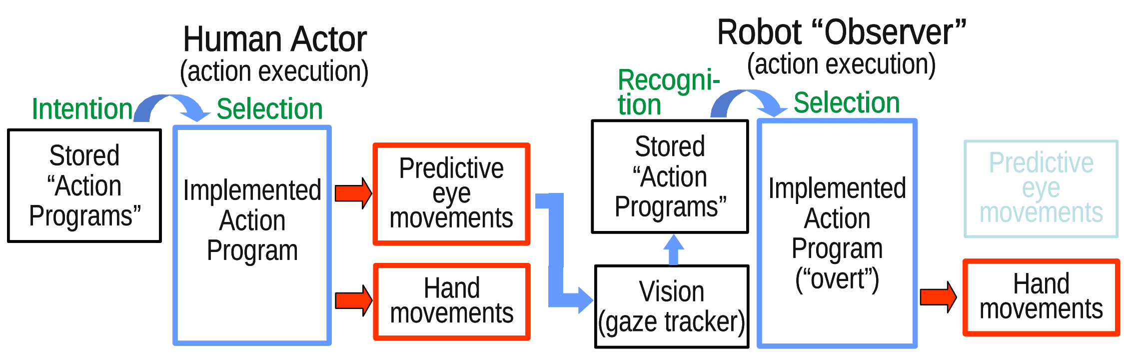

Fig. 2. A robot observer that tracks the actor's gaze movements to make predictive robotic hand movements.

3 Experimental Setup

The basic setup is illustrated in Fig. 3. A user is seated in front of the manipulation area holding the bar (2 x 2 x 8 cm) and asked to make contact to two target points in particular order before returning to the home position (contact position 0). The location of the contact points and the dimension of the work area are identical to [2]. All movements take place on a vertical frontal plane, termed the work plane, which is located between the user and the faceLAB gaze-tracking hardware [4].

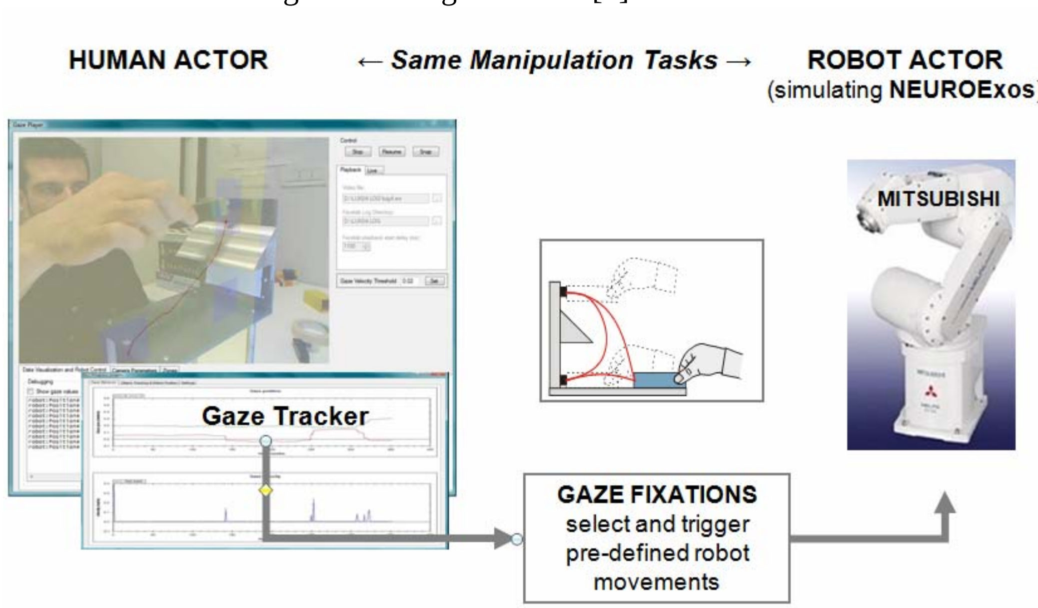

Fig. 3. Gaze tracking setup. The gaze tracker software implements the gaze tracking algorithm that is used to select and trigger the robot motions. The main window of the software shows the gaze fixation point (in red) in the work plane (yellow overlay). The faceLAB tracking system (not shown) is in front of the user and the work area.

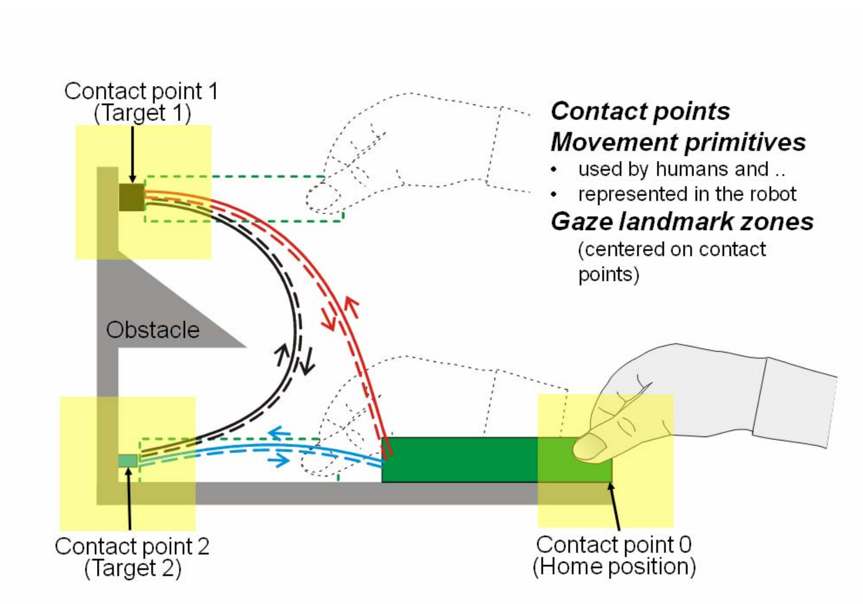

Fig. 4. Task movement primitives, contact points, and gaze landmark zones (yellow squares)

3.1 Gaze tracking

The gaze trcker used in this work is the faceLAB, shown in Fig. 5. This system is a complete solution which includes hardware and software. The main hardware components are a pair of Sony FCB - EX480B Sony cameras, accompanied by two infra-red pods for tracking in Precision Gaze configuration and for tracking in night or in a dark environment. A Stereo-Head holds the two cameras, and is designed to be adjustable for different user configurations; it also holds two IR pass lens filters which ensure consistent ambient lighting on the face. These filters can also be used for subjects wearing glasses in order to reduce the reflections on the lenses from a monitor or large projector screen.

Fig. 5. Components of the FaceLab system.

This system is able to detect: the head pose, Gaze direction, Eyelid movement, Pupillometry, World Model interaction, Image Features and both head and eye measurements have an associated quality level. The system is able to detect the head and eye movement with a reference system located between the two cameras, as shown in Fig. 6. In function of the parameter detected by cameras and the features constructed by using the dedicated software it is possible to reconstruct the head movements in 3D and the configuration assumed by the subject.
Moreover it is possible using the software dedicated to create a world model. In this way the real experimental platform can be reproduced in a virtual environment and verified in real time during the experiment.

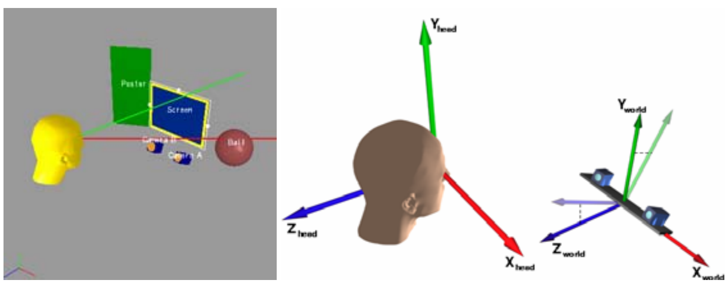

Fig. 6. The FaceLab world model.

The faceLAB system was configured for precision mode which allowed head tracking and individual eye tracking. Although the faceLAB system tracks many head and gaze variables, for the purposes of this experiment, we used the gaze fixation point which is computed automatically by faceLAB as the intersection between the unified gaze ray and the work plane, defined relative to the world coordinate frame. The current gaze fixation point and trail is overlaid on the video display (from the application's camera). Left and right eyeball gaze angles are also tracked to compute the gaze angular velocity.

3.2 Tasks

Each user is asked to perform four tasks shown in Fig. 7. Tasks 1 to 3 are performed with 4 repetitions. Task 4 is performed for a total of 30 contacts with the order of the contacts determined by the user.

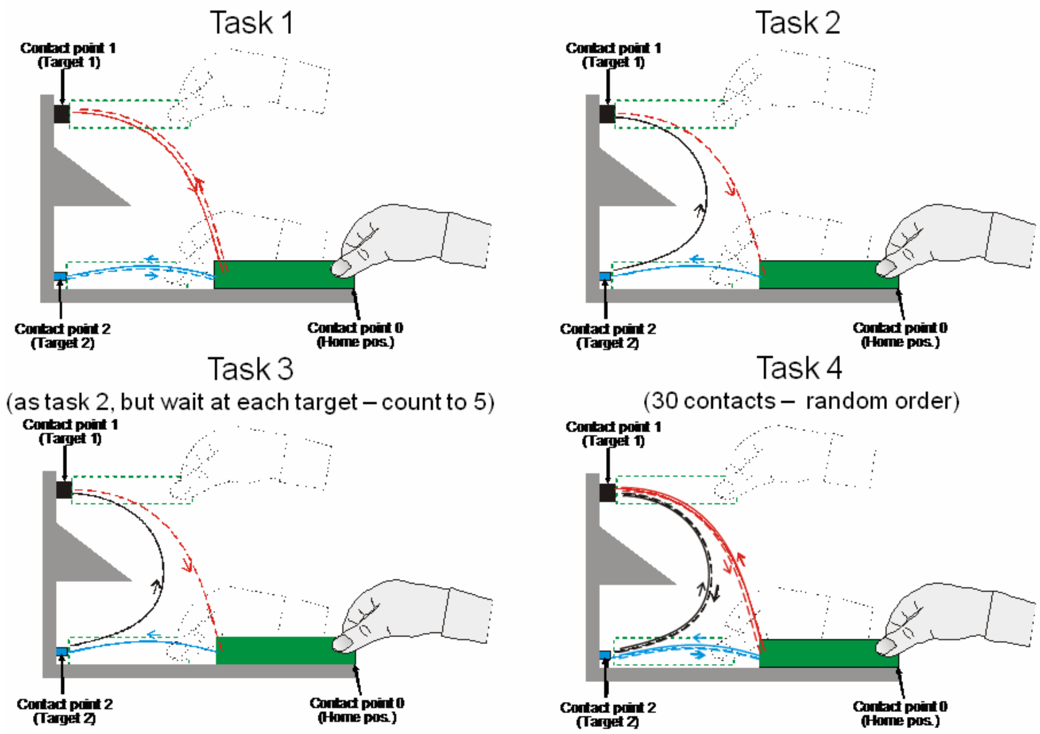

Fig. 7. Four goal-directed tasks performed by the user.

3.3 General Procedure

Four right-handed male subjects were asked to perform the four tasks. A faceLAB forward-only head model was then made for each subject. Before the beginning of each task, the user is given instructions on what to do with the object. Each is advised to keep their head stationary while doing the task since the viewable area in precision faceLAB mode is limited. With the exception of Task 3, no time limit was given for each segment of the task. The only instruction was to move the object in a natural manner.

3.4 Gaze Tracker: software implementation details

Custom software was written to accept faceLAB data in real-time via the network, and to track and display the gaze fixation point over the video of the user performing the task. The software records the incoming faceLAB data and video which can later be played back. The software also implements the gaze interface algorithm that sends commands to the Mitsubishi robot that performs the motion primitives as inferred from the user's gaze behavior.
The software performs the following functions:

Real-time gaze tracking

Gaze data from the faceLAB system is received via UDP packets over a LAN connection. All faceLAB variables are received and recorded by the gaze tracker using the provided C++ library functions of the faceLAB client tools SDK. For tracking the gaze fixation point, the gaze-object intersection data as computed by faceLAB and left and right eye gaze angles are tracked in real-time. Gaze quality is also monitored and used to filter noisy or inaccurate data. If the total gaze quality (the sum of the left and right eye gaze quality value) falls below 5 (out of a maximum of 6), the current gaze-object intersection data is not included in the gaze queue. This gaze queue is implemented as a circular buffer to minimize memory requirements. Subsequent processing utilizes gaze values obtained from the queue. The gaze angular velocity is computed as the difference between the current and last gaze eyeball pitch and yaw angles (retrieved as the top 2 values in the queue). Gaze angles from only one eye, the one with higher gaze quality, is used for computing the gaze angular velocity.

Robot motion triggering

The Mitsubishi industrial robot was programmed with all the task movement primitives. To initiate movement, the robot is controlled via numbered commands that are sent via a TCP-IP socket; upon receipt of such a command the appropriate motion primitive subroutine executes. The primary source of motion commands is a state machine which implements the gaze algorithm and is described in section 4.

Video processing and recording

Since faceLAB does not provide video recording, we added a calibrated USB camera to record a video of the task trials. The calibrated camera allowed us to project on the video an overlay of the work plane, gaze zones and the gaze fixation point. It also serves as a vision-based sensor for tracking the bar held by the user. To accomplish all of this, a DirectShow custom filter graph was used for

- processing each video frame (640 by 480 pixels at 24 Hz) for color object tracking (when enabled).
- video recording to Avi file. It also creates time stamps for each video frame which is used during gaze playback synchronization.
- displaying a transparent overlay of the work plane and gaze zones.
- playing back video from an avi file. During playback object tracking and gaze zone overlay is also available. Synchronization messages are sent to the main form thread to allow the gaze playback thread to ensure gaze and video playback are in sync.

Vision-based object tracking

For the purpose of tracking the bar held by the user a vision based object tracking was used. The color tracker uses the camshift [5] algorithm found in the OpenCV library[6]. The reported position in the video image is used for data visualization.

Data visualization

The following variables are plotted in real-time versus the faceLAB frame number.

- gaze fixation point (x, y) in meters
- gaze velocity in radians/second
- object position in video image (pixels)
- robot position state

Gaze playback

Gaze playback is useful for analyzing gaze behavior and consists of video playback with a synchronized gaze trail overlaid. This is done by reading the faceLAB database created during the real-time recording session. A thread is spawned for a function that reads values from the database and performs the same processing as in real-time gaze tracking. This thread synchronizes itself with the video playback thread to ensure that gaze point displayed actually corresponds to the recorded moment in time as shown in the video. Gaze playback uses the same visualization tools and updates the robot state machine.

4 Gaze based control

We develop a simple robot control algorithm that utilizes the gaze behavior of the user. The method consists of using the landmark zones and gaze velocity. When the user's gaze enters one of the zones and the gaze velocity is below a threshold we trigger a transition on the robot state machine. The robot state machine encodes all the possible movements for the tasks. The zones are 6 x 6 cm in size while the gaze speed threshold is set to 0.01 radians/second. Fig. 8illustrates the gaze fixation criteria used for determining when to trigger the robot.

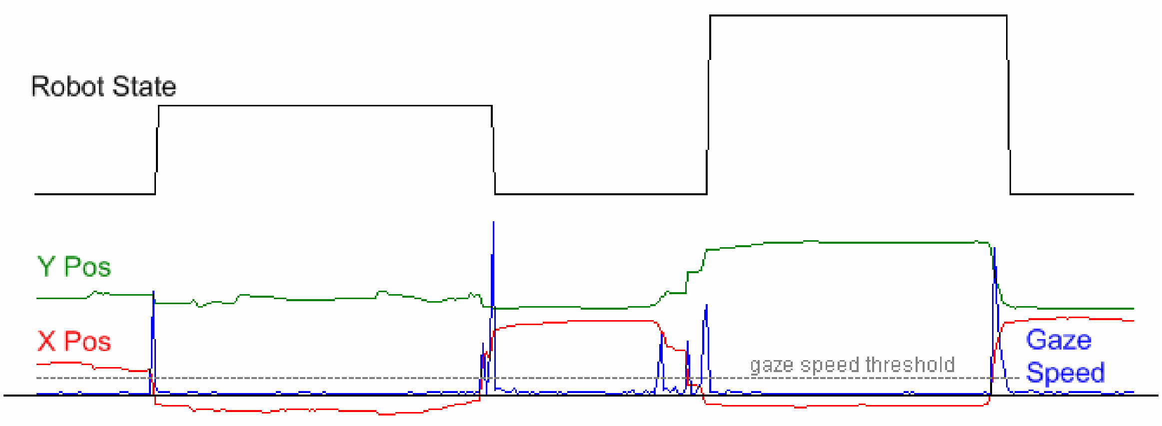

Fig. 8. Criteria for selection of gaze fixation which are used in the state machine as triggers for state transition. The gaze fixation point must be located inside a landmark zone and the gaze speed is below a given threshold. State transitions occur when the criteria is met.

5 Results

Fig. 8 is taken from the real-time graph of Task 4 from subject 4 and illustrates the typical gaze behavior in relation to object movement. The robot transitions are overlaid on both graphs and show when robot transitions occur. The top figure shows robot transitions in relation to object movement while the bottom figure shows the robot transitions in relation to gaze position and velocity.

We define two measures, *Ta* and *Tb*, and are shown in Fig. 8 as directed arrows. *Ta* measures the time when robot transition occurs relative to the object leaving the current contact point. *Tb* measures the time when robot transition occurs relative to when the object reaches the next contact point. Right pointing arrows indicate positive values which means the robot transition occurs in advance of the object event (contact point entry or exit) while left-pointing arrows represent negative values which mean the robot transition is delayed in relation to the object event of interest.

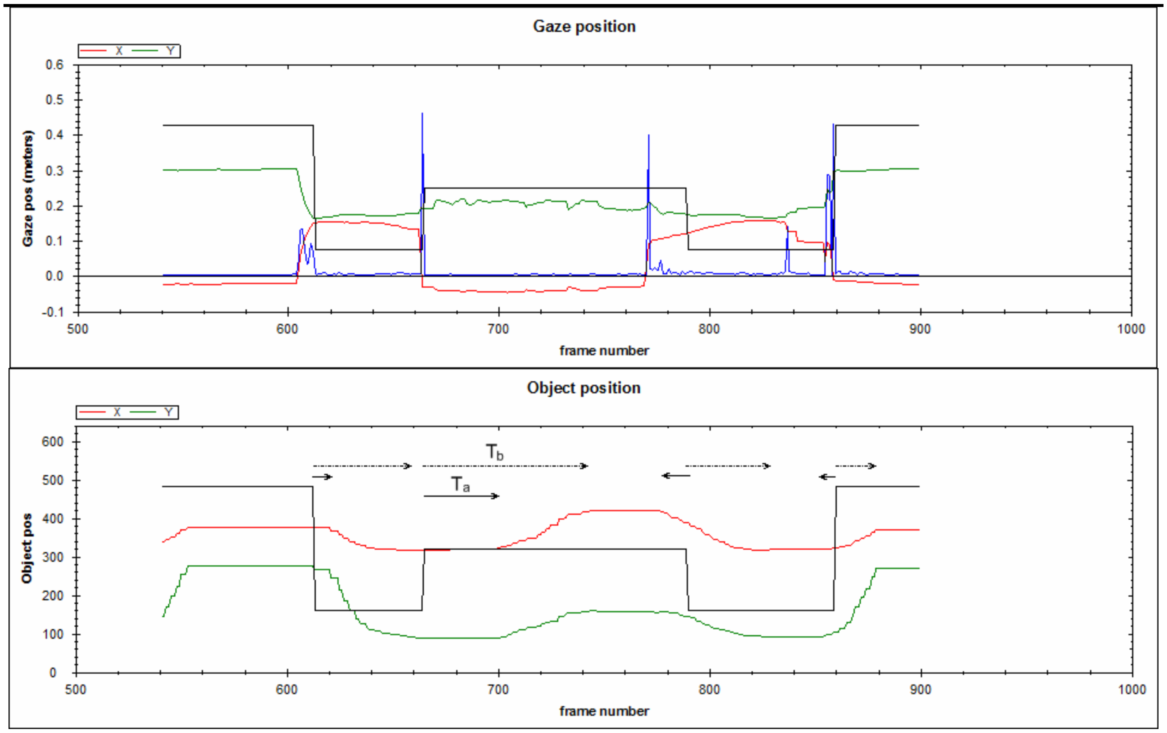

Fig. 9. Typical robot transitions in relation to object position (top) and gaze position and gaze velocity (bottom). The robot transition always occur before the object reaches the next contact point (dashed arrows, Tb) and sometimes before the object exits the current contact point (solid arrows, Ta). Robot transitions are in black while x,y positions are in red and green, respectively. Gaze velocity is in blue. Frame rate = 60 Hz.

5.1 *Task Prediction Performance*

Table I shows that on average it gaze enters the next landmark about the same time the object leaves the current contact point but the coupling between these events is weak as evidenced by the large standard deviation. An exception is task 4, where the user tends to fixate at the next destination before moving his hand. This may be because the user is deciding first where to go next unlike the first 3 tasks where the motions are pre-defined.
As can be seen from Table II, the robot transition occurs always before the object reaches the next contact point. Since robot transitions are triggered when the gaze enters the next landmark zone, this result is consistent with the reported gaze behavior in [2] in which the gaze enters the landmark zone where contact is about to be made approximately 1 sec before.
The algorithm is useful as far as predicting where the next contact point will be but cannot be relied upon to exactly tell when the hand (object) will leave the current contact point.

5.2 *Implementation Related Issues*

The measured times, Ta and Tb, include the delays introduced by the system implementation. Such delays are expected and are part of the motivation for using predictive control methods.

Also, measurement errors are present in the faceLAB data and the vision-based object tracking. In the case of the gaze data, a quality-level is included by faceLAB and is used to filter noisy gaze data.

TABLE I
Robot transition vs. current contact point exit (Ta)

Task Num	Average	Standard Deviation
1	3.31	16.41
2	-1.86	19.88
3	2.00	11.72
4	-7.85	13.96

Units are in faceLAB frames. 1 frame = 1/60 Hz.

TABLE II
Robot transition vs. next contact point entry (Tb)

Task Num	Average	Standard Deviation
1	57.87	12.77
2	56.87	22.94
3	45.58	37.43
4	40.95	16.06

Units are in faceLAB frames. 1 frame = 1/60 Hz.

Acknowledgment

The authors thank Pericle Salvini, Fernando Tello Gamarra, Luigi Manfredi, Ekahwahyu Susilo, and Eliseo Stefano Maini for valuable assistance conducted during this work.

Bibliography

[1] Land M., Mennie N., Rusted J. “The roles of vision and eye movements in the control of activities of daily living”. *Perception* 28(11):1311-1328, 1999.
[2] R. S. Johansson, G. Westling, A. Backstrom, and J. R. Flanagan, "Eye-hand coordination in object manipulation," *J Neurosci*, vol. 21, pp. 6917-32, Sep 1 2001.
[3] J. R. Flanagan and R. S. Johansson, "Action plans used in action observation," *Nature*, vol. 424, pp. 769-71, Aug 14 2003.
[4] "Seeing Machines, FaceLAB face and eye tracking system," 2007.
[5] G. R. Bradski, "Computer vision tracking for use in a perceptual user interface," *Intel Technology Journal*, vol. 2, 1998.
[6] I. Corporation, "Open Source Computer Vision Library Reference Manual," 2005.