Integrating Vision for Human-Robot Interaction

Benjamin R. Fransen
Beyond Robotics, Inc.
3508 Newport Ave.
Annapolis, Md 21403
benjamin.fransen
@beyondrobotics.biz

Wallace E. Lawson
Naval Research Laboratory,
Code 5515
4555 Overlook Ave., SW,
Washington, DC 20375, USA
ed.lawson@nrl.navy.mil

Magdalena D. Bugajska
Naval Research Laboratory,
Code 5515
4555 Overlook Ave., SW,
Washington, DC 20375, USA
magda.bugajska@nrl.navy.mil

Abstract

Human-robot interaction necessitates more than robust people detection and tracking. It relies on the integration of disparate scene information from tracking and recognition systems combined and infused with current and prior knowledge to facililtate robotic understanding and interaction with humans and the environment. In this work we will discuss our efforts in the development and integration of visual scene processing systems for the purpose of enhancing human robotic interaction. Our latest efforts in integrating 3D scene information for the production of novel information sources will be discussed and demonstrated. We show the integration of facial pose and pointing gestures to localize the diectic gesture to a single point in space. Additionally, we will discuss our efforts in integrating Markov logic networks for high level reasoning with computer vision systems to facilitate scene understanding.

1. Introduction

The goal of Human-Robot Interaction (HRI) research is to build robotic systems capable of human like interactions [8]. Recent advances in sensor technologies have enabled development of robust people detection and tracking systems. The wide range of human interactions and the dynamics of those interactions necessitates integration of people detection and tracking methods in a principled manner. To facilitate integration, both low level computer vision systems and high-level reasoning must be designed to facilitate a bidirectional flow of information.

Integration presented in this paper demonstrates two forms of knowledge transfer that begin the production of a full bidirectional information passing system. First, we will discuss our work in producing 3D scene information where scene elements are recognized and tracked in 3D. Through this work we will discuss and demonstrate information passing between tracking systems and the integration of gaze and body pose tracking for the purpose of isolating diectic gestures to a single point in space. Second, we will discuss our effort in integrating Markov Logic Networks (MLN) with vision systems to provide high-level logical inference.

We begin by describing the details of our tracking systems in Section 2. In Section 3 we will present integration and application of the tracking systems in the context of deictic gesture recognition. We will then discuss high-level reasoning via Markov logic networks in Section 4 followed by conclusions.

2. Tracking Systems

To facilitate visual tracking we have integrated a team of tracking algorithms that can track at multiple scales and exchange information for error handling/correction, tracker initialization and integrated data analysis. For tracking people at long range a full body tracking algorithm has been developed. The full body tracking system models clothing color using an adaptive particle filter for tracking one or more people moving within a scene. At closer, conversational, distances body pose and position are tracked using an optical flow based face tracker and an anatomically based arm and head tracking system. After a brief description of our robotic hardware, each of the three systems we are using will be discussed individually starting with the full body tracker, followed by the body pose tracker and the face tracking system.

2.1. Sensor System

The tracking systems have been fielded on a Mobile-Dextrous-Social (MDS) robot [4]. The MDS has both range and Electro-Optical (EO) imaging as shown in Figure 1. The range camera is located in the forehead with EO imagers located in the independently moving eyes. The MDS's range camera used for this system is a Swiss

978-1-4244-7030-3/10/$26.00 ©2010 IEEE

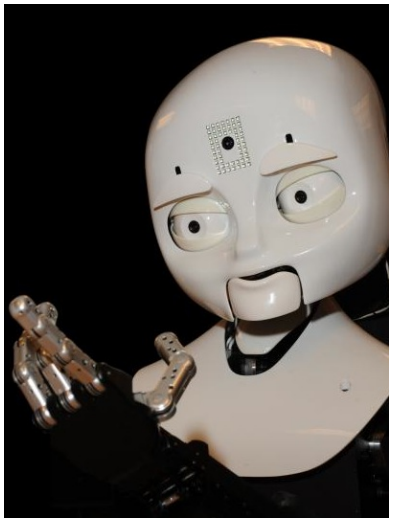


Figure 1. Close up of sensor systems. Each eye contains a Point Grey FireFly camera and the Swiss Ranger time-of-flight sensor is located in the forehead.

Ranger time-of-flight sensor. The Swiss Ranger produces a 176x144 image with a 47.5°x39.6°field of view and a maximum range of 7.5m. Point Grey FireFly cameras, located in each eye, produce 320x240 pixel images at 60 frames/second.

2.2. Full Body Tracking

Full body tracking is produced using a hue based upper and lower body clothing model. At long ranges, the TOF sensor is not effective for recovering spatial information. For this reason, full body tracking is performed using color information only. This allows tracking of distant people at ranges of 30 feet and beyond.

The system begins tracking by detecting a person in the robot's field of view. A boosted cascade of Haar-like features [17] provide an image region for each person. The region is split into an upper and lower half to account for differences between a shirt and pants. The shirt and pants regions are then compared separately with surrounding background to develop a maximal discriminative color histogram for both the upper and lower body. A particle filter is then used to estimate position and size for bounding regions of each person. The system has been tested with up to five people in real time tracking experiments.

To facilitate interaction with additional vision modules, the person tracking system outputs indexed particle images that capture the positions for particles tracking each person. Particle images are utilized in conjunction with depth images to estimate person location and upper/lower body orientation.

2.3. Body Pose Tracking

The body pose tracking system is responsible for identifying and tracking the user's face, hands and elbows, and extracting three-dimensional coordinates for each. We use depth information from the TOF ranger in combination with color images for both identification and tracking to provide accurate body pose recovery and achieve robustness in the presence of multiple people.

2.3.1 Merging Range and Luminance Images

Range and color images are independently taken by the color camera and TOF sensor. Before we can use them in an integrated fashion the data must be registered at the pixel level.

The transformation from range to color image coordinates is recovered through a least square registration method described by Horn et al [11]. Once the coordinate transformation is recovered, a projection matrix is used to project range into the luminance image as follows.

A luminance image I is a one-dimensional function defined at pixel locations $\vec{u}_i = \begin{bmatrix} u_i \\ v_i \end{bmatrix}$. Similarly, a range image is a one-dimensional function defined at each pixel $\vec{x}_j = \begin{bmatrix} x_j \\ y_j \\ z_j \end{bmatrix}$. Using a pinhole camera model, correspondence between the range image and luminence image is recovered.

A pinhole camera projective model P with focal lengths (F_u, F_v) and optical center (C_u, C_v) projects points from 3D to image coordinates as

$$u_i = \frac{F_u\, x_j}{z_j} + C_u,\ v_i = \frac{F_v\, y_j}{z_j} + C_v. \qquad (1)$$

Renumbering corresponding points in the the range and luminence image, we now have a correspondence between points imaged by the TOF sensor and pixels in an image

$$(u_i, v_i) = P(x_i, y_i, z_i).$$

P can be recovered through stereo camera calibration [16] or generated for any web camera through freely available calibration toolboxes [20].

2.3.2 Body Pose Person Detection

In the person detection phase, the system has two modes. When the full body tracker is actively tracking people, regions identified by the particle image, but within 12 feet of the camera, are searched for frontal faces. If the full body tracker is not actively tracking anyone, the system searches its entire field of view for faces within 8 feet of the robot.

body pose tracking begins with face detection. In conjunction with particle images provided by the full body tracker, we utilize a simple heuristic to determine when to detect the face. The particle images generated by the full body tracker are

Once a face is found, skin color is learned by creating a color histogram of the ellipse enclosed in the detected face region. For face detection, the algorithm used is based on the boosted cascade object detection algorithm as described by Lienhart and Maydt [17] and implemented in the OpenCV open source computer vision library [3].

To faclilitate interaction with additional vision modules, the body pose person detection system outputs head, hand, and elbow positions. These position are used for object interactions recognition through proxemics and for initialization and verification of additional trackers.

2.3.3 Pose Tracking

Once the system is initialized with a face location and skin color histogram, it needs to track the skin colored regions corresponding to the face and hands across frames. This problem of human tracking has been widely studied by researchers, resulting in numerous tracking algorithms. Many algorithms use complex human models and statistical frameworks to fit the models to observed data [19, 21]. However, these models must be learned or roughly estimated, and computing model parameters for many degrees of freedom is very computationally expensive. Since we are interested in a real-time tracking system, we use a human body model in combination with skin color. We assume that the three regions of interest, head and two hands, all contain approximately the same hue, and we track those regions using an approach that extends the work of Argyros [1].

The algorithm proposed by Argyros utilizes multiple hypothesis labeling to track skin-colored regions in each frame. When a skin-colored region is not explained by a hypothesis, a new hypothesis is created. Similarly, when a hypothesis is no longer explained by any observed points, it is removed from the list of tracked hypotheses. Argyros's algorithm deals with cases where one region explains two hypotheses, such as when two skin-colored arms touch or intersect. However, it does not handle the case where two skin-colored regions explain one hypothesis, such as when one arms skin colored region is split by a watch into two skin colored regions. In this case, the original algorithm assigns a hypothesis to one of the two regions and creates a new hypothesis for the remaining region, instead of treating the two regions as one. In addition, the original algorithm could be distracted by other persons in the background whose skin colored regions might appear contiguous in the camera view.

To address these problems, our algorithm extends Argyros algorithm in two ways. First, we allows more than one region to explain a hypothesis, by considering two blobs, B_1 and B_2, as one if $d(B_1, B_2) < \tau$ for a distance metric d, and a threshold τ. The distance metric τ is calculated from the 3D Euclidean distance between the closest pixels from two blobs. Second, three dimensional positions are used to segment pixels by comparing against a 3D body model.. Thus, if a second person is behind or beside the tracked person, the second persons skin colored pixels would be ignored if they exist within regions that can not be explained physiologically.

When the face is first detected, it is assigned a hypothesis. In subsequent iterations of the algorithm, a newly found skin colored region is classified as either the left arm, right arm, or if both are currently tracked, it is discarded. During initialization, the classification of a region as left or right hand is done based on the relative horizontal position. If one hand is already being tracked, the new region is assumed to be the remaining hand. When the head track is lost, the system reenters the detection phase.

In the case where the whole forearm is visible, such as when the person is wearing a short-sleeve shirt, a 3D body model disambiguates the hand from the elbow at ends of the ellipse. The tracking algorithm runs in real-time at speeds of up to 28 frames a second on a 3.2 GHz Pentium IV system. A sequence of tracked images from the body pose tracker is shown in Figure 4 C.

2.4. Face Tracking

Face tracking is performed by a system we refer to as FLOAT, [7], which stands for Fast Linear Optical Apperance Tracker. FLOAT tracks faces and objects by comparing color images with a learned models in real time. For face tracking, a person's face is recorded in 3D using the registered color and range data and then compared against a video stream to recover both 3D position and pose information. A training image with multiple views is shown in figure 2. In facial tracking tests the systems runs at speeds greater than 60 frames per second with accurate results for both position and orientation returned for distances less than ten feet from a camera at 320x240 resolution.

To recover the position and orientation of a face, a stored model must be registered with an object located in the video stream. The problem of image registration consists of finding, for each image I_i in a sequence, a geometric transformation between the scene as seen in the first image I_0 and as seen in I_i, parameterized by a vector $\vec{\mu}(i)$. At time 0 an object template consisting of a set $\mathcal{R} = \{\vec{x}_i(0)\}$ of 3D points that are acquired with the help of a depth camera as mentioned previously. We center the template at the origin to help minimize numerical error when applying transformations by the motion model,

Figure 2. FLOAT training data.

$$\vec{f}(\vec{x}(0), \vec{\mu}(t)) = R \begin{bmatrix} x(0) \\ y(0) \\ z(0) \end{bmatrix} + \vec{T} \quad (2)$$

$$= \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x(0) \\ y(0) \\ z(0) \end{bmatrix} + \begin{bmatrix} T_x \\ T_y \\ T_z \end{bmatrix} \quad (3)$$

One method employed for image registration algorithms is gradient descent. One group of gradient-descent-based algorithms, called compositional algorithms, assume that warps form a group with the composition operation. Baker [2] introduced the term inverse compositional image alignment (ICIA), an especially efficient compositional algorithm, and showed its mathematical equivalence to previous popular gradient-descent-based approaches. Recent work by Buenaposada and Muoz [12] gives a very readable introduction to ICIA. While most uses of gradient descent algorithms have been planar, (*e.g.* [9]), we track in 3D.

One distinction for work presented here from previous work is the motion model employed. Previous work utilizes an incremental motion model based on the small angle approximation, eg. [10]. Because we intend to track large rotation changes, we provide a derivation for recovering a full rotation matrix without restricting each incremental change by the small angle approximation. This is done for tracking of rapidly moving objects and to facilitate future use of this algorithm in object recognition experiments where detecting objects at large angle changes is beneficial. Additionally, no depth information is needed at run time. All depth data is collected during training. This eliminates the introduction of noisy depth data and reduced frame rates needed to perform stereo depth recovery or capture depth data from active systems, such as a Swiss Ranger, during run time.

One straight forward choice for the parameter vector $\vec{\mu}$, based on equation 3, is then
$\begin{bmatrix} T_x & T_y & T_z & a & b & c & d & e & f & g & h & i \end{bmatrix}$.

We apply our motion model in an ICIA framework as follows. Estimate $\vec{\mu}$ at each timestep by least-squares:

$$O(\vec{\mu}) = \sum_{\vec{x} \in \mathcal{R}} \left(\vec{I}(\vec{f}(\vec{x}, \vec{\mu}), t_0) - \vec{I}(\vec{x}, t) \right)^2.$$

Since a rigid object can't change size, for our purposes "scaling" must be interpreted as a change in z coordinate: after optimizing over $a \ldots i$ but before normalizing them, we can set

$$T_z(t + \tau) = |R| \, T_z(t).$$

We can remove g, h, i and T_z from the vector we put through least-squares, reducing the dimensionality of the optimization.

Now

$$\vec{\mu} = \begin{bmatrix} T_x & T_y & a & b & c & d & e & f \end{bmatrix},$$

$$\frac{\partial \vec{u}}{\partial \vec{x}}(\vec{x}) = \begin{bmatrix} \frac{F_u}{z} & 0 \\ 0 & \frac{F_v}{z} \end{bmatrix}$$

and

$$\frac{\partial \vec{f}}{\partial \vec{\mu}}(\vec{x}_i(0), \vec{\mu}(0)) = \begin{bmatrix} 1 & & \vec{x}_i(0)^T & \\ & 1 & & \vec{x}_i(0)^T \end{bmatrix}.$$

Note that despite the fact that we're now explicitly writing some elements of $\vec{\mu}$ as functions of others, for runtime efficiency we retain the assumption that all variables are independent. Since by assumption f_z is independent of (the new) $\vec{\mu}$, the third row $\frac{\partial f_z}{\partial \vec{\mu}}$ of $\frac{\partial \vec{f}}{\partial \vec{\mu}}(\vec{x}_i(0), \vec{\mu}(0))$ becomes zero, so we drop it and $\vec{f}$ becomes a two-dimensional function $\begin{bmatrix} x_i(t) \\ y_i(t) \end{bmatrix} = \vec{f}(\vec{x}_i(0), \vec{\mu}(t))$.

We follow [12] in performing a small fixed number of least-squares iterations per frame, using the most recent frame I_k as an approximation of I_t for

$$t \in \{k, k + \tau, k + 2\tau, \ldots\}.$$

As a gradient descent algorithm, face tracking requires an initial location and orientation. Face locations can be provided by body pose tracking, face detection, or face recognition algorithms. Alternatively, the system can scan the scene, however, this generally requires a greater computational burden than the alternatives.

Results for the face tracking system are demonstrated in Figure 3.

This module outputs position and orientation of the head. Location is returned in millimeters and rotations are returned in degrees. Output is utilized for non-verbal communication, such as head nods, verification of body pose tracking results and gaze recognition.

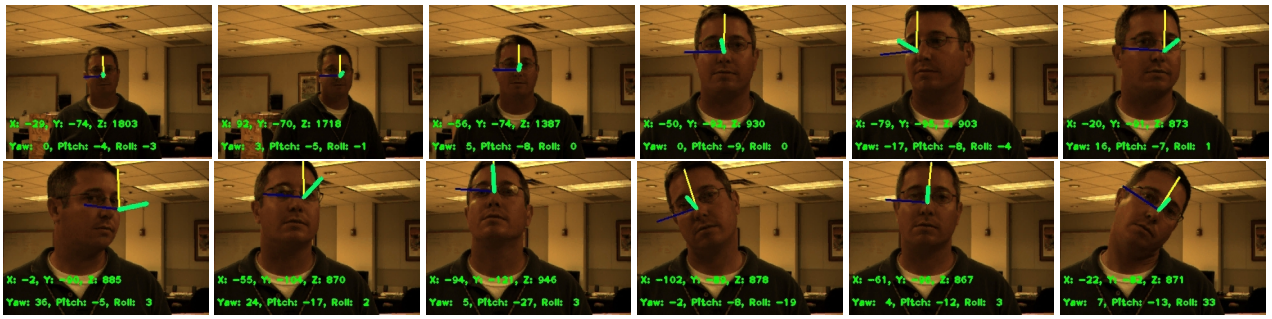


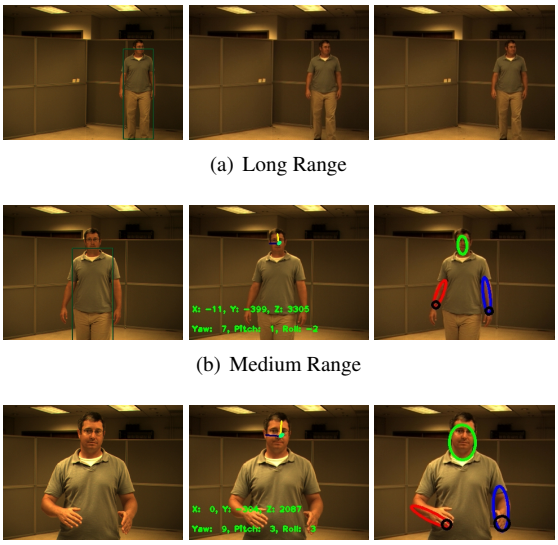
Figure 3. FLOAT tracking results.

3. Integrated Tracking and Applications

While each of the tracking systems can run in a standalone fashion, the simultaneous operation of tracking systems producing and consuming information in concert with one another provides a more versatile and robust system. We discuss integration work and applications of that integration in this section.

3.1. Integrated Tracking

We demonstrate the integration of the systems described in Section 2 by tracking a person at multiple ranges in Figure 4. At long ranges, the full body tracker detects a person as shown in row (a). Because the figure is out of accurate tracking range for the face and body pose tracking neither is tracking as shown in row (a). As the person approaches, as shown in row (b) the full body tracker has passed person location information to the other trackers which have begun tracking. Although the face tracker is actively tracking, tracking results at ranges in excess of ten feet often times have error and the track is not dependably accurate. For such cases, the redundant data from the body pose head track can be used to validate/invalidate the face tracking results. At a closer range, as shown in row (c), the face tracker is accurately tracking as is the body pose tracker. The full body tracker has invalidated the track due to occlusion of the lower body caused by being too close of proximity to the robot. By handing off information and tracking at appropriate scales the tracking systems are capable of tracking over a large range of distances with appropriate information being available for each range.



(a) Long Range

(b) Medium Range

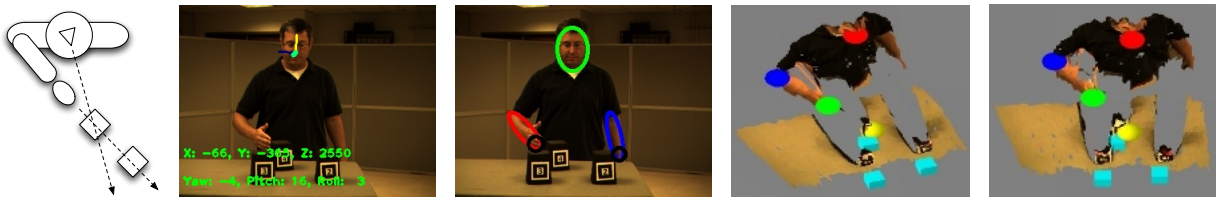
(c) Close Range

Figure 4. Person tracking activity as the human approaches the robot.

3.2. Applied Tracking

By recovering 3D scene information, integrated tracking results can be utilized to extend capabilities of the system. We demonstrate one such capability here in the form of interpreting deictic reference.

While following a three dimensional line from the elbow to the hand out into the scene is applicable to many situations, ambiguity can arise as to which of many objects a person is pointing. Such a situation is illustrated in Figure 5 (a) where two objects could be referenced based on the hand's pointing direction. To disambiguate these situations, hand and face direction are integrated to provide a location in space where both the hand and face are pointing. Figure 5 (d) and (e) show two different views of the 3D scene while showing locations of tracked objects. The right elbow and hand are labeled in blue and green, the face has a red sphere about the nose and the joint hand and face deictic gesture is illustrated in yellow. Pointing location is estimated by finding the intersection of the hand pointing direction with a plane that bisects the face. By integrating hand and face pointing direction we are able to narrow down the deictic gesture to a single point and reference that location specifically.



(a) (b) (c) (d) (e)

Figure 5. Through combined body and facial pose tracking deictic references are recovered as a point in space rather than lines (as shown in diagram (a)). Image (b) shows the face track, (c) shows the body pose track, and (d,e) show the recovered pointing location from different perspectives using a 3D viewer. In addition to body tracking fiducial tracks are shown as blue squares in front of the numbered markers used by the fiducial system.

4. Scene Understanding

Tracking and object recognition alone will not facilitate scene understanding. To develop a fully working system, we need the ability to deal with conflicting data, to invalidate erroneous hypothesis and lost tracks, while simultaneously integrating prior knowledge and bringing together multiple modalities. To facilitate these needs, we have begun evaluating Markov Logic Networks.

4.1. Markov Logic Networks

Logical representations of AI are extremely adept at handling complexity and making inferences about the real world. The difficulty with logical representations is that in some cases information cannot be stated without uncertainty. The power of the Markov Logic Networks [5, 18] lies in it's ability to fuse a logical representation with statistical uncertainty.

Our logical representation includes people, actions, and physical objects. We define relations between objects using first order logic, and use Markov networks to assign a weight to each relation in terms of relative importance. In formal inference, if one formula is violated, the world has zero probability of occuring. Weighting eases this restriction by saying that if a relationship is violated, it is still possible, just less likely.

Formally, a Markov Logic Network (MLN) M is a set of formulas F_i and associated real valued weights w_i (paired $L = (F_i, w_i)$). Combining L with a set of constraints C gives an MLN $M_{L,C}$.

The probability of a set of ground predicates X is defined as

$$P(X = x) = \frac{1}{Z} \exp\left(\sum_{i=1}^{m} (w_i n_i(x)\right) \qquad (4)$$

where $n_i(x)$ is the number of true groundings of the world under the assignment x. Z is a normalizing constraint, to ensure that the values sum to 1.

The MLN with Maximum A Posteri (MAP) inference finds the most probable state, given evidence. In the case of human interaction, the evidence is provided by prior knowledge about the subject and actions, as well as information from the sensors. The MLN uses all of the available information to return the state of the world that is most likely. For deictic gesture resolution, the physical object to which the subject is referring would be inferred as part of this world.

MLN learning can be performed either generatively or discriminatively. For work presented here, we utilized discrminitive learning. The difference between the two forms of learning is that discriminative learning knows a priori which predicates will be evidence and which will be queries. Generative learning assumes that all predicates may be queries. We refer the interested reader to [18] for further details on learning weights. All learning and inference procedures are packaged together in the Alchemy package[15], which we make use of in this paper.

4.2. MLN Modeling

We use MLNs to make inferences about the real world, based on who we are observing and what we have seen them interact with in the past. For training, several example scenarios were used describing human knowledge about people and their interactions.

In the evaluation phase, the state of the world is provided by vision systems. For example, a face recognition engine will tell us the identity of a person[13], while a body pose tracker [6] and fiducial tracker[14] will tell us about interactions between people and objects.

In this initial work, we show the applicability of the MLNs to scene analysis by evaluating interactions in a simple laboratory environment. We define the following predicates for this domain. **id** refers to a unique tracking id that is assigned to every newly tracked individual or an object.

1. **Ben(id)**, **Ed(id)**, **Magda(id)** - A person being tracked is identified as one of the known individuals.

2. **Hammer(id)**, **Screwdriver(id)**, **Laptop(id)** - An object being tracked is identified as a laboratory equipment.

3. **ReferencesHammer(id)**, **ReferencesScrewdriver(id)**, **ReferencesLaptop(id)** - A person has made a deictic reference to a physical objects in the scene.

4. **InteractsWithHammer(id)**, **InteractsWithScrewdriver(id)**, **InteractsWithLaptop(id)** - A person has interacted with (picked up or touched) a physical object in the scene.

Some individuals may have a strong prior towards performing an action. For example, we may observe Ben fixing robots all the time, whereas Magda only uses the computer. When we detect an individual enter the scene, we can infer what they will do based on their identity.

Using predicates, this is expressed as

- $Ben(x) \Longrightarrow FixesRobot(x)$
- $Magda(x) \Longrightarrow UsesComputer(x)$

In order to permit the MLN to learn about what individuals will perform each action, our predicates include all possible combinations of individuals and actions.

When an individual interacts with an object as defined by their relative proximity, this provides an indication about what objects they will refer to in the future. For example, if an individual has recently interacted with the screwdriver, they will have an increased likelihood to reference the hammer.

We capture this using predicates

- $InteractsWithHammer(x) \Longrightarrow ReferencesScrewdriver(x)$

Once again, we include all possible combinations in order to permit the MLN to learn.

Finally, if we know what a person is doing, we can predict what they will reference. For example, if you are fixing the robot, you will be more likely to reference the screwdriver

- $FixesRobot(x) \Longrightarrow ReferencesScrewdriver(x)$

Again, we use all possible combinations to permit learning.

5. Conclusions

The wide range of human interactions and their dynamics requires principled integration of people detection and tracking methods. We presented a methodology for integrating low level visual processing systems with each other and with high level reasoning. Our framework addresses the dynamics of information processing across the whole range of interaction space from intimate to public in addition to the incorporation of prior knowledge and high level reasoning through Markov logic networks. We employ this system for real time human-robot interaction tasks and have demonstrated it here for person tracking and deictic gesture resolution.

Acknowledgements

This work was supported by grant number N0001407WX20452 from the Office of Naval Research to Greg Trafton. The views and conclusions contained in this document should not be interpreted as necessarily representing the official policies, either expressed or implied, of the U. S. Navy.

References

[1] Antonis A. Argyros and Manolis I. A. Lourakis. Real-time tracking of multiple skin-colored objects with a possibly moving camera. In *European Conference on Computer Vision (ECCV 2004)*, volume 3, pages 368–379, 2004.

[2] S. Baker and I. Matthews. Equivalence and efficiency of image alignment algorithms. In *In IEEE Conference on Computer Vision and Pattern Recognition (CVPR01)*, 2001.

[3] G. Bradski, A. Kaehler, and V. Pisarevsky. Learning-based computer vision with intel's open source computer vision library. In *Intel Technology Journal*, May 2005.

[4] Cynthia Breazeal, Michael Siegel, Matt Berlin, Jesse Gray, Rod Grupen, Patrick Deegan, Jeff Weber, Kailas Narendran, and John McBean. Mobile, dexterous, social robots for mobile manipulation and human-robot interaction. In *SIGGRAPH '08: ACM SIGGRAPH 2008 new tech demos*, pages 1–1, New York, NY, USA, 2008. ACM.

[5] Pedro Domingos, Stanley Kok, Hoifung Poon, Matthew Richardson, and Parag Singla. Unifying logical and statistical ai. In *AAAI'06: Proceedings of the 21st national conference on Artificial intelligence*, pages 2–7. AAAI Press, 2006.

[6] Benjamin Fransen, Vlad Morariu, Eric Martinson, Samuel Blisard, Matthew Marge, Scott Thomas, Alan Schultz, and Dennis Perzanowski. Using vision, acoustics, and natural language for disambiguation. In *HRI '07: Proceedings of the ACM/IEEE international conference on Human-robot interaction*, pages 73–80, New York, NY, USA, 2007. ACM.

[7] B.R. Fransen, E.V. Herbst, A. Harrison, W. Adams, and J.G. Trafton. Real-time face and object tracking.

In *Intelligent Robots and Systems, 2009. IROS 2009. IEEE/RSJ International Conference on*, pages 2483 – 2488, oct. 2009.

[8] M. A. Goodrich and A. C. Schultz. Human-robot interaction: A survey. *Foundations and Trends in Human-Computer Interaction*, 1(3):203–275, 2007.

[9] Gregory D. Hager and Peter N. Belhumeur. Efficient region tracking with parametric models of geometry and illumination. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(10):1025–1039, 1998.

[10] M. Harville, A. Rahimi, T. Darrell, G. Gordon, and J. Woodfill. 3d pose tracking with linear depth and brightness constraints. *iccv*, 01:206, 1999.

[11] Berthold K. P. Horn. Closed-form solution of absolute orientation using unit quaternions. *J. Opt. Soc. Am. A*, 4(4):629, 1987.

[12] Luis Baumela Jose M Buenaposada, Enrique Munoz. Efficient illumination independent appearance-based face tracking. *Image and Vision Computing*, In Press, 2008.

[13] Behrooz Kamgar-Parsi, Wallace Lawson, and Behzad Kamgar-Parsi. Recognizing faces like humans. *SPIE Newsroom: Electronic Imaging and Signal Processing*, February 2010.

[14] H. Kato and Mark Billinghurst. Marker tracking and hmd calibration for a video-based augmented reality conferencing system. In *Proceedings of the 2nd International Workshop on Augmented Reality (IWAR 99)*, San Francisco, USA, October 1999.

[15] S. Kok, P. Singla, M. Richardson, and P. Domingos". The alchemy system for statistical relational ai. Technical report, Department of Computer Science and Engineering, University of Washington, Seattle, WA, 2005.

[16] K. Konolige. The sri small vision system. http://www.ai.sri.com/konolige/svs/.

[17] Rainer Lienhart and Jochen Maydt. An extended set of haar-like features for rapid object detection. In *International Conference on Image Processing*, 2002.

[18] Matthew Richardson and Pedro Domingos. Markov logic networks. *Mach. Learn.*, 62(1-2):107–136, 2006.

[19] Hedvig Sidenbladh and Michael J. Black. Learning the statistics of people in images and video. *Int. J. Comput. Vision*, 54(1-3):181–207, 2003.

[20] A. Whitehead and G. Roth. The projective vision toolkit, 2000.

[21] Ying Wu, Gang Hua, and Ting Yu. Tracking articulated body by dynamic markov network. In *ICCV*, pages 1094–1101, 2003.