Conferring human action recognition skills to life-like agents

(Abbreviated title: Action Recognition Skills for Agents)

Luc Emering, Ronan Boulic, Daniel Thalmann

LIG - Computer Graphics Lab., Swiss Federal Institute of Technology,

Lausanne, CH-1015 Switzerland

{emering, boulic, thalmann}@lig.di.epfl.ch

Abstract

Most of today's virtual environments are populated with some kind of autonomous life-like agents. Such agents follow a pre-programmed sequence of behaviours that exclude the user as a participating entity in the virtual society. In order to make inhabited virtual reality an attractive place for information exchange and social interaction, we need to equip the autonomous agents with some perception and interpretation skills. In this paper we present one skill: human action recognition. By opposition to human-computer interfaces (HCI) that focus on speech or hand gestures, we propose a full-body integration of the user. We present a model of human actions along with a real-time recognition system. To cover the bilateral aspect in human-computer interfaces, we also discuss some action response issues. In particular we describe a motion management library that solves animation continuity and mixing problems. Finally we illustrate our system with two examples and discuss what we have learned.

Introduction

The possibility of interaction with synthetic characters is a decisive aspect in the engagement of a participant as an active member in reactive inhabited virtual environments (Pausch et al., 1996). Because of its intuitive nature, a gesture-based interface is appealing for both novice and experienced users (Rubine, 1991). Various techniques have been proposed in order to interpret the participant's motion. Newby uses statistical classification of hand postures in order to identify the American Sign Language (Newby, 1994). Roy et al. exploit a connectionist approach in order to recognise an arm gesture from a list of arm mimes (Roy et al, 1994). Darrell et al. extract the location of visible end effectors from the participant image (Darrell et al., 1994). They derive high level input from the detection of specific postures or gestures.

Most often, the motion capture of a participant is either limited to a small part of the body such as a hand or an arm, or restrained by visibility constraints. Gavrila & al. aim at identifying the full human posture by analysing multiple-view frames (Gavrila et al., 1996). They have also addressed the problem of movement recognition for the two upper limbs (Gavrila et al., 1997). However, this goal was achieved in a post-processing phase. By comparison, the alternate motion capture approach based on magnetic sensors allows close to real-time identification of full participant mobility (Semwal et al., 1996), (Molet et al. 1998). Furthermore, the magnetic sensor technology now has higher usability thanks to a partially wireless device. We take advantage of the magnetic sensors in order to provide the joint value input. Our contribution answers the participant's action recognition problem by proposing a hierarchical model of body action, a real-time recognition algorithm and a motion mixing method for action responses. The system is illustrated with an immersive virtual shop environment, populated by two autonomous agents, the shopkeeper and a friend called 'Mary', and the performers' avatar. The example's aim is to illustrate how action recognition can be seamlessly integrated within the behaviours of the autonomous agents in order to create an interactive social scenario. In a second example, the interface uses action recognition in order to control the performer's interaction with a virtual environment, by body and camera displacements, object grasping and light control. In previous work, we used action recognition for a simplified fighting training application (Emering et al., 1997).

Application Fields

A good human-computer interface is one that allows the user to forget the interface and concentrate on the task. The interface manipulation should become part of the user reflexes. To reach this goal, the interface has to heavily use the participant's' a priori knowledge and

daily experiences. Therefore, we propose full-body human action recognition as an important component of multi-modal interfaces. Some application fields that we have already experienced or foresee for our system are the following:

1) performing tasks in a simpler way

One goal in virtual reality (VR) is to simulate real world situations. As a consequence, the interface has to facilitate the real world interaction paradigms. However, reproducing every single detail of an interaction is not necessarily a desirable goal. Grasping an object is such an example. It is relatively easy to copy a performer's hand motions onto virtual hands. Even collision detection between objects and the virtual hands can be mastered (Boulic et al., 1996) (Rezzonico et al., 1995). Nevertheless, experience shows that grasping a virtual object is still more difficult than in real life. The problem is the poor quality of the feedback loop. Neither can we manage realistic 3D-vision sensation, nor do we have realistic force feedback devices. We think that grasping an object with detailed finger control is not necessarily desirable. Instead of asking the user to perform a precision grasp, we ask the user to perform a grasp gesture. This action is recognised by the system and the object snaps to his/her hands. Of course, in real life, objects generally do not behave like magnets, but this can be an acceptable compromise between the realism of a task performance and the convenience of the interface. Other interaction tasks that can be advantageously simplified through action recognition events are walking, sitting or simply looking around by turning the head. In most VR systems, such actions are triggered by some hand posture recognition events. We claim that such actions can be more naturally triggered through a full-body action recognition system.

2) context dependent reactive agents

Imitation, in the sense of copying with modifications or reformulating, is a key aspect in simulating intelligence for synthetic agents. Famous examples are robots in MUDs (Multi-User-Dungeon), for example the robot called Julia in (Foner, 97). The principle consists of analysing text strings and finding some corresponding answer in a dictionary or to reformulate the input text. A similar approach can be applied to the behaviour of synthetic agents. In our virtual shop testbed, we let the seller imitate the users' pointing to the merchandise, in order to incite the life performer to confirm the choice. Another example that we are currently developing consists of an interactive dance performance between a live performer and an autonomous dancer. Most of the time the autonomous dancer simply copies the performer's dance motions, possibly with some amplitude modulation. Whenever the autonomous dancer recognises a specific dance action, it suspends the imitation and plays a pre-recorded keyframe sequence. Such context dependent behaviors can be useful in interactive pedagogical applications, for example to train people (Rickel et al., 1998).

3) semantic modulation

Human gestures are often performed in parallel with speech. They can either modulate the emotional context of the dialog or drastically change its semantics. If used as a stand-alone channel, gestures are self-contained, semantically meaningful units. A method for isolating the emotional motion components of gestures is described in (Amaya, 1996). We did not develop any testbed that integrates the recognition system with a speech synthesizer, but we think it is an interesting issue for interactive environments focusing on dialogues between performers and agents (Cassell et al., 1994).

4) interface for human embodiment and environmental controls

Rather than using action recognition for communicating with another agent, we can use such a system to interact with an environment by controlling some parameters such as: the virtual camera positions, displacement parameters of the performer's embodiment, or changing the behaviour of objects. Further in this paper, we present an interactive walk-through environment that reveals the possibilities and also some pitfalls in the design of such applications.

Previous Work

Currently there are two main approaches for designing a full-body action recognition human-computer interface. The first directly connects to the sensors' raw data stream. This is generally the case for video image based sensing installations. The ALIVE system (Maes et al., 1995) applies vision routines in order to extract the performer's figure from the 2D image background and label the salient points of the contour. The recognition algorithm itself uses space and time separable template patterns. The KidsRoom (Bobick et al., 1997) moves beyond just measuring salient body points. It combines sensor output with contextual information derived from the story in order to recognise individual and group actions, e.g. 'staying in a group'. Another interesting fact in ALIVE is that the user has an embodiment in the virtual scene - the extracted video figure - whereas in the KidsRoom, user embodiments are absent.

Motion-based recognition from video images can be decomposed into describing a spatial pattern *(where* there is motion) and how the motion evolves. An extension proposes 'motion-energy images' and 'motion-history images' as a two-component version of a temporal

template. The recognition consists of matching the templates with stored models of views of known actions (Bobick and Davis, 1997).

The other approach consists of first reconstructing a 3D model from the sensor data. This is typical for systems with a small set of high precision sensors such as data gloves, magnetic or optical motion capture systems. Here the data is mapped to anatomical joint angles (Molet et al., 1996) prior to the recognition stage. The advantage is that the sensor dependent layer can be replaced without necessarily affecting the recognition system.

The underlying recognition algorithms that can be found in literature are various. Hidden Markov Models, HMM, (Yanghee Nam et al., 96) are among the most popular, as the probabilistic approach seems to adapt well to complex human actions that are difficult to clearly define. HMM suffer however from a large number of parameters that have to be adjusted either by training or by some a priori assumptions. Also it is not clear, due to a lack of publications, whether HMMs could reasonably scale to higher dimensional data input spaces, such as human bodies, without conflicting with real-time constraints. Other popular methods are neural networks (Looney, 1997). The critical decision lies in the choice of an adequate network type and the parameter adjustment functions. However, neural networks are known for robust classification in presence of noise. Finally, there are statistical methods. They are widely used because of their lightweight computation and implementation facilities.

Human Actions

The purpose of human activity is to achieve common or individual goals. In order to identify these goals by a software process, we need to divide human activity into smaller units, actions

that can easily be categorised by an algorithm. As human activity is driven by multiple motivations such as biological needs, rational decisions and social goals, we limit our model to the action types having the following characteristics:

- *goal-oriented*. The initiation of the action is motivated by a unique final goal. The action has a rational character, i.e. its goal is the result of a conscious decision. The goal is often underlined by a final permanent body posture or a characteristic set of constraints in the Cartesian space (detailed later in this paper). We do not consider actions of biological or physiological character such as fidgeting or balance gestures. We consider fidgeting as a noise whose amplitude is not big enough to have a significant impact on the action specifications. Finally, at the current level of development, we do not consider actions of social character, although they are interesting for higher level behavioural interactions. Social actions can be modelled as a composition of rational actions. For example, 'having a glass of wine with somebody' can be decomposed into 1) grasping the glass 2) clink glasses 3) drink and 4) put the glass down. Actions of social character have, typically, many subgoals, which is in contradiction with our previous requirement of unique goal-oriented actions.

- *finite duration*. Time is a good criterion for delimiting the set of actions that we are interested in. We consider only actions of short duration, that is, actions of a few seconds. Actions of social nature have a longer lifetime, in the order of minutes and more. For periodic actions, such as walking, we claim that the definition of one motion cycle is sufficient. During the recognition phase, periodic actions will be recognised each time a cycle of the periodic motion is identified.

- *postures & gestures*. Actions are expressed through the performer's body configuration, either in joint or Cartesian space. Postures are given by a permanent body configuration (absolute values at a given time) by opposition to gestures (variations over time).

- *involved body parts.* Actions do not necessarily involve the complete body. A set of bodyparts can be sufficient to perform an action. A body part is one of the following: upper or lower arm or leg, a hand, a foot, the neck, the head, the pelvis, the abdomen or the thorax (Table 1). Reducing an action definition to a set of bodyparts does not necessarily imply a loss of action relevant-information.

- *parallelism*. Sometimes, actions are performed in parallel rather than sequentially. Giving a phone call with a mobile telephone while walking is considered as two different actions. The execution of actions can be sequential, overlapping or parallel, i.e. the 'phoning' action may start and end at any time before, during, or after the walking action. We also state that distinguishing parallel actions in an unambiguous way implies the use of non-intersecting sets of body parts for both actions.

On the basis of these observations, we define an action as follows, with a BNF-like (Backus-Naur-Form) notation:

Action ::= <posture> | <gesture> | <gesture*posture> | <constraint> | <action + constraint>

where '*' means 'followed by'

'+' means a parallel evaluation of both operands

An action can be defined by a gesture followed by a final posture. An example is ‘sitting down’ which is decomposed into the ‘sitting down‘ gesture of the body and the permanent sitting posture. Starting postures are not taken into account, as we are mainly interested in goal-oriented actions expressed with a well-defined final body state. Another reason is that for an action such as ‘sitting down’, a large number of starting postures can be considered. We are not interested in the performer’s state at the beginning of an action.

Besides postures and gestures, we also consider actions that are defined by constraints. In short constraints allow the defining of actions such as ‘holding an object’. Constraints are either verified or not. They produce a Boolean result at each time step and are evaluated in parallel to an action. This concept will be detailed further in this paper.

Action Data

In order to preserve the mobility of a human skeleton in a body model, one has to handle a large number of degrees of freedom (DOF), around 40-80, excluding hands and feet. Our current model has 68 DOF for the main skeleton (cf. Figure 1).

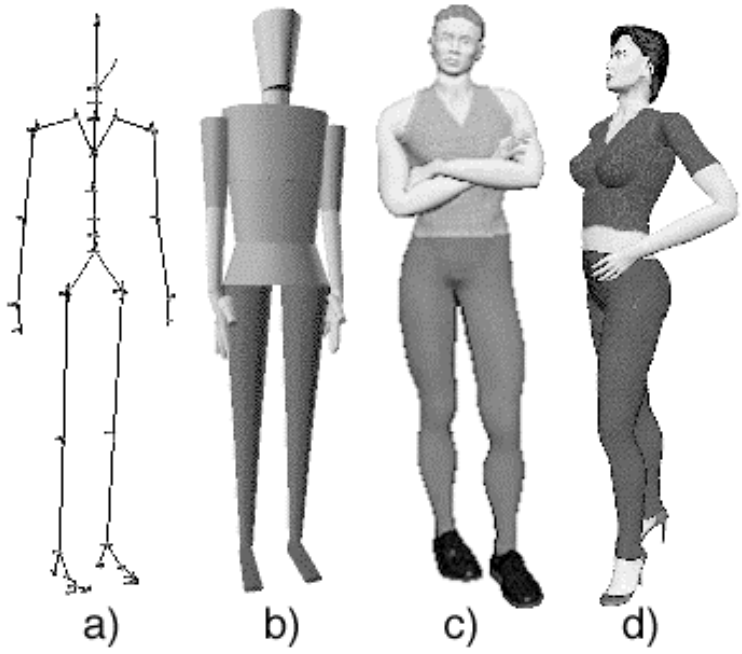

Figure 1 Several display choices for a humanoid

Tracking and analysing such an amount of data is a time-consuming computation process that needs to be optimised. Our approach consists of defining a hierarchy of three data levels. At the top level, we consider the centre of mass (CoM). The CoM is a rough criterion that carries information regarding the global body configuration and motion direction. At the second level, we consider end effectors (EE): hands, feet, head and spine root. We include the spine root because it reveals the position of the upper body relative to the lower body. End effectors represent a remarkable compression of body information. Indeed, one can decode a human action solely by looking at the position evolution of these strategic body references (cf. Figure 2). A slightly larger set of body landmarks has been used in (Hodgins, 1995) to compare real and simulated motions. At the lowest level, we consider the skeleton joint angles (JOINT). The JOINT angles are the most detailed body information source available.

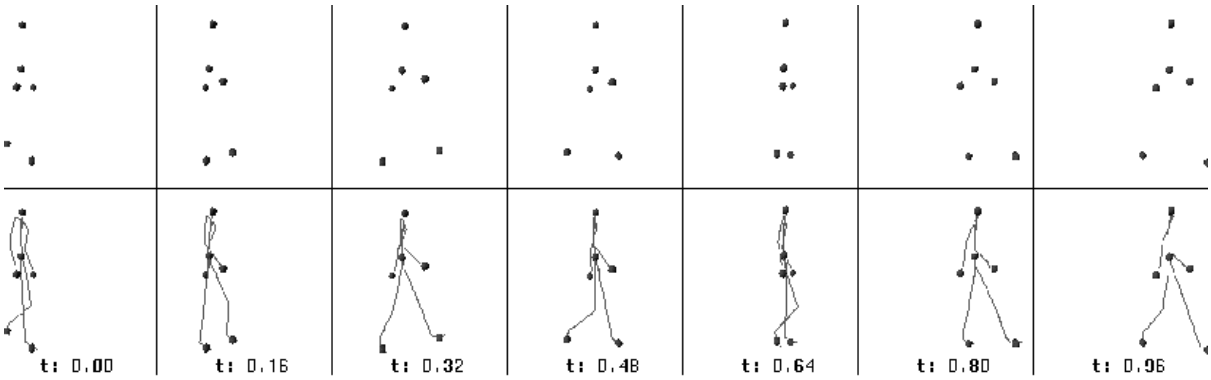

Figure 2 Showing strategic body locations during a ‘walking’ motion (cf. our mpeg files at http://ligwww.epfl.ch/~emering/AAI_JOURNAL)

Cartesian data and joint angle variations depend heavily on the anatomical differences of the live performers. We propose to normalise all Cartesian data with the body height of the performer. Indeed, statistical studies have measured high correlation between the body height and the major body segment lengths (Kroemer, 1990). Choosing adequate reference coordinate systems also matters. We use three coordinate systems, shown in Figure 3: a Global, a Body and a Floor Coordinate System (in short *GCS, BCS, FCS*). The BCS is attached to the spine base of the body and its up axis is aligned with the main spine direction. The FCS is located at the vertical projection of the spine root onto the floor level and reuses the GCS Up axis.

An approximate CoM position is derived from the positions and the masses of the geometric volumes attached to the body skeleton (cf. Figure 1b). The CoM is referenced in the FCS because the height to the local floor is determinant. Consider for example the two postures: ‘standing’ and ‘lying on the floor’. If the CoM were expressed in the BCS, the postures would not be distinguishable by the CoM. The EEs, except the spine root, are referenced in

the BCS. Indeed the spine root is referenced in the GCS, so that it indicates global body location and movements.

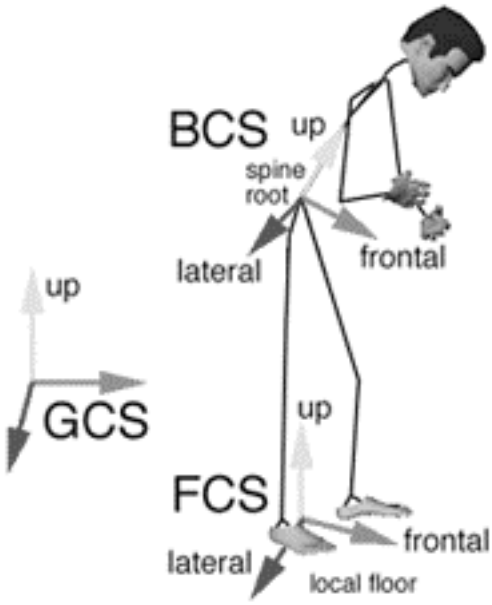

Figure 3 Reference coordinate systems (Global, Floor and Body CS)

Action Primitives

Based on the three information levels, CoM, EE, and JOINT, we introduce the concept of action primitives. An action primitive is an elementary unit used for the action specifications. We define posture, gesture and constraint primitives:

- Posture primitives are of the form (body location, 3D position) and (JOINT, angle value) where body location is either the CoM or one EE. Explicitly specifying 3D positions and angle values is tedious. Therefore, we use a specify-by-example paradigm. For each posture, we record a prototype and store it in a database. The system then extracts the posture primitives directly from these prototypes.

- Gesture primitives have the form (body location, velocity direction) where ‘body location’ denotes either the CoM or one EE. The ‘velocity direction’ is a discretised value describing the average motion direction: up, down, forward, backward, left, right. Its value is set whenever the velocity norm is above a normalised threshold along one of the FCS, BCS or GCS frame vectors. The threshold is fixed empirically. It delimits voluntary motions from noise, caused by fidgeting, for example. Additionally a *not_moving* value for the ‘velocity direction’ specifies a still CoM or EE. The head end effector is a special case as we consider rotation directions rather than velocity directions. This is useful to express messages like ‘yes’ and ‘no’. The rotation direction is determined by analysing the average rotation of the look-at vector. For gesture primitives, we don’t use JOINT level information because pattern recognition on joint angle evolution functions is incompatible with real-time constraints.

- Constraint primitives are of the form (EE_i, EE_j, f) with $i \neq j$, and ‘f’ a distance comparison function. A constraint primitive is either verified or not, depending on the result of ‘f’ applied to both end effectors. In the simplest case, ‘f’ is a function that compares the relative distance between the end effectors against a threshold value. An example is the ‘holding an object’ action: both hands are constrained to a constant distance, independently of the current posture or gesture. More sophisticated constraint functions are possible, e.g. checking the relative orientation between end effectors or checking whether an end effector is inside a bounding volume. So far, we applied only distance constraints.

Action Model

Postures are defined by a Boolean expression of posture primitives. As we use posture prototypes that are defined at JOINT level by default, the system can automatically extract the missing data of CoM and EE levels. As a consequence, postures are always defined by posture primitives of the three levels. Any posture can be expressed as a logical 'and' expression of posture primitives. When inserting posture prototypes into the posture database, it is possible to define the relevant body parts of a posture. The set of proposed body parts is listed in Table 1. Additional DOF for body positioning are not listed because they are not used in the model.

Body Part	# of joints	# of DOF
BODY_HEAD	1	3
BODY_NECK	2	6
BODY_THORAX	2	5
BODY_ABDOMEN	3	5
BODY_PELVIS	1	3
BODY_L_THIGH	2	6
BODY_R_THIGH	2	6
BODY_L_LEG	1	2
BODY_R_LEG	1	2
BODY_L_FOOT	3	4
BODY_R_FOOT	3	4
BODY_L_U_ARM	3	7
BODY_R_U_ARM	3	7
BODY_L_L_ARM	1	2
BODY_R_L_ARM	1	2
BODY_L_FIST	1	2
BODY_R_FIST	1	2
Total	31	68

Table 1 Set of body parts and number of degrees of freedom

Gestures are defined explicitly by a Boolean expression of gesture primitives. All gesture primitives are optional and can be combined with logical ‘and’ and ‘or’ operators. Some examples of gesture definitions:

'Walking forward’ = ((spine_root, forward) AND (left foot, forward))
OR ((spine_root, forward) AND (right foot, forward))

‘Sidewalking right’ = ((CoM, nop) AND (spine root, right) AND (left foot, right))
OR ((CoM, not_moving) AND (spine root, right)
AND (right foot, right))

‘Say yes’ = (head, up) OR (head, down)

‘Welcome’ = (right hand, up) AND (right hand, right)
AND (left hand, up) AND (left hand, left)

‘Wave right hand’ = ((right hand, left) AND (left hand, not_moving))
OR ((right hand, right) AND (left hand, not_moving))

‘Pointing right hand right’ = (right hand, right) AND (right hand, up)
AND (left hand, not_moving)

‘Pointing right hand forward’ = (right hand, forward) AND (right hand, up)
AND (left hand, not_moving)

‘Sit down’ = (CoM, backward) AND (spine, backward) AND (spine, downward)

‘Get up’ = (CoM, forward) AND (spine, forward) AND (spine, upward)

Table 2 Some gesture specifications

All the definitions have been empirically set and can be more or less constrained, by adding more or less ‘not_moving’ primitives. We did not list symmetrical gestures, e.g. left-sided gestures, ‘walking backwards’ etc.

By using action primitives based on CoM, EE and JOINT, our action model has a hierarchical structure with five definition levels and one parallel branch for the constraints.

		Level	Level description	
Action	**Gesture**	1	Centre of Mass (velocity)	constraint(s)
		2	End Effector (velocities)	
	(final) Posture	3	Centre of Mass (position)	
		4	End Effector (positions)	
		5	Joint values	

Table 3 Hierarchical action model

Action Recognition

In the near future, more and more motion capture systems will provide real-time animation possibilities of a performer's avatar. In order to be independent of this technology progress, we base our recognition system entirely on the virtual avatar body rather than on raw sensor data. With this approach, other motion capture systems, such as optical ones, can be used. Of course, a corresponding anatomical converter, from the sensor data to the virtual body model, has to be provided with that motion capture system.

Currently we use a magnetic motion capture system with a maximum of 16 sensors. A real-time anatomical converter (Molet et al., 1996) maps the raw sensor data onto a virtual skeleton model. The resulting avatar is used as input to the recognition system (cf. Figure 4).

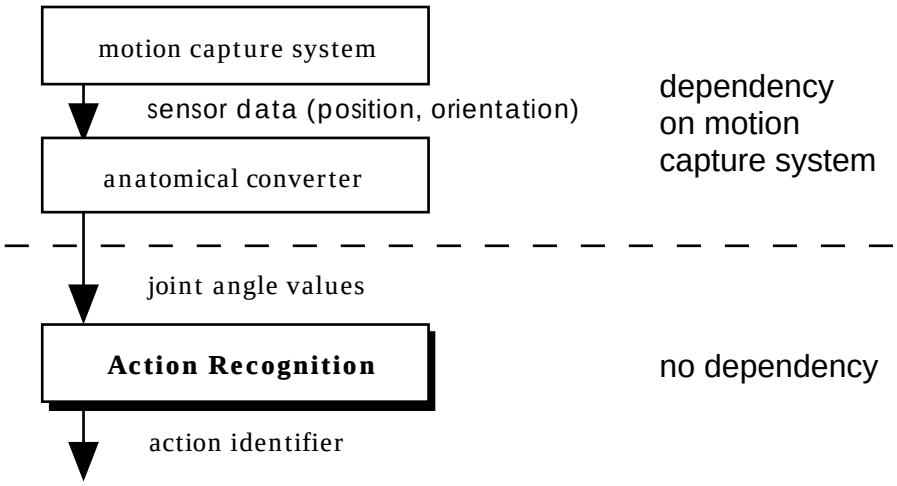

Figure 4 Data flow from the sensors to the recognition system

Recognition algorithms

The action recognition system is based on the action model presented in the previous section. The algorithm starts by initialising a candidate database with all defined actions: the Candidate Action Set (CAS). Then it proceeds sequentially with a candidate selection at each of the five levels of Table 3. Candidates that don't correspond to the performer's current action are eliminated from the CAS. If at any level the CAS becomes empty, the system returns an 'unknown action' message. Candidates that have no definition for a given level, bypass this level during the candidate selection, i.e. they stay in the CAS.

The recognition algorithm takes advantage of the hierarchical data structure of the action model. It is a constant balance between the number of action candidates and the amount of data per candidate. At the higher levels, the CAS is large: we apply rough selection criteria such as CoM and EE, with low computational cost. As the CAS passes through the selection levels, its size shrinks whereas computation costs increase. At the lower levels, computation

costs are higher because they deal with JOINT space entities. The multi-level approach allows real-time performances.

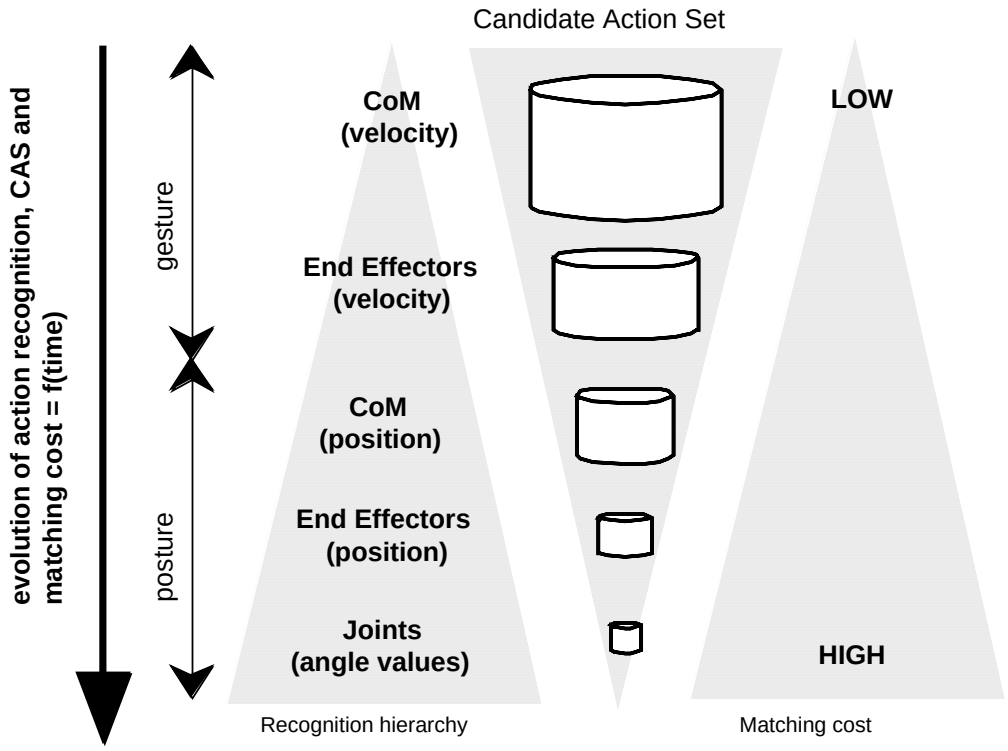

Figure 5 Action candidate elimination scheme

1) Gestures (Levels 1 & 2).

Here we compute the current gesture primitives of the avatar. The primitives are substituted into each action's Boolean gesture expression. Those actions whose expression results in a *TRUE* value remain in the CAS. The others are discarded immediately to avoid unnecessary computation at lower levels.

2) Postures (Levels 3, 4 & 5)

Here we compute the 3D position of the CoM. For all the actions in the CAS, the squared distance between their CoM and the avatar's CoM is calculated. We call *Min* and *Max*, the

respectively smallest and largest squared distances. Actions remain in the CAS exclusively if their distance is smaller than a selectivity radius *R* given by:

$$R = Min + (1-S)*(Max-Min)$$

Equation 1 Posture selection

S is a normalised selectivity parameter within [0,1]. We apply the same algorithm at levels four and five with possibly a different, empirically fixed, selectivity factor. The selectivity factor regulates how aggressively the candidates are removed at each level. The dimension of the Cartesian vector varies at the three levels: 3D for the CoM, 18D for the EEs (concatenation of six end effector 3D positions) and 68D for the joints (68 degrees of freedom of our body model). Note that this algorithm always selects at least one posture among the posture candidates. This is true even if the avatar's posture doesn't correspond to any of the postures defined in the database. This artefact may or may not be desirable, depending on the compromise of interaction mis-interpretation and its ignorance. If misinterpretation happens too often, a simple remedy is to add some neutral postures to the database, for example a 'stand still' posture. Those postures are generally not imposed by application needs but they help avoid aberrant posture selections. An alternative is to include a cut-off threshold in Equation 1. The problem of this solution is to fix the threshold. We discuss some improvements of the posture selection algorithm further on.

3) Constraints

Constraints are evaluated in parallel to the gesture and posture levels. This consists of applying the constraint functions to the performer's current body configuration. Each constraint results in a Boolean value.

Simultaneous actions can be detected as long as they act on complementary body parts. ‘Making a phone call while walking’ is such an example. We define the phone call action by a posture involving the right arm and the neck (levels four and five). 'Walking’ is defined by a gesture of the lower body.

Recognition Performance

Performances have been measured with an action database of thirty-five actions. The database includes 13 actions with all the levels, six actions with only gesture levels, and 16 actions with only posture levels. The resulting average recognition processing time is 1.2 ms (R4400 CPU, 200 MHz) with a very small standard deviation (less than 0.1ms) (cf. Figure 6). The processing time is not necessarily equal to the recognition time. For example, recognising a ‘walking’ action requires waiting for approximately one quarter of a walking cycle until the gesture primitives are verified. In practice, this means a ~0.2s delay until ‘walking’ is recognised. For postures and constraints however, the recognition time equals the processing time.

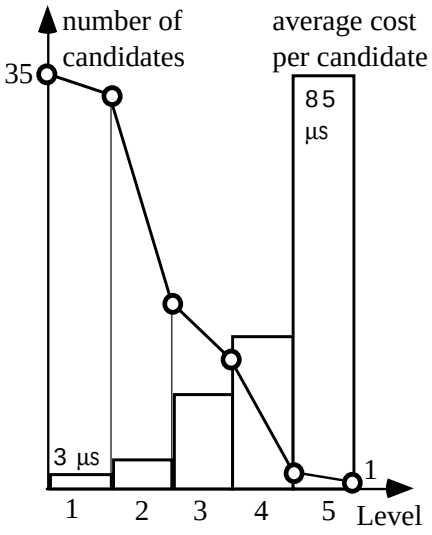

Figure 6 Recognition processing cost analysis

The robustness (cf. Figure 7) is evaluated on four performers whose total heights range from 1.65m to 1.85m. The tested action set consists of four gesture-only actions, 27 posture-only actions and 16 full-level actions. The test has the following structure: stand still, perform one action, then stand still again. We imposed this motion sequence to our test performers in order to have a comparable basis.

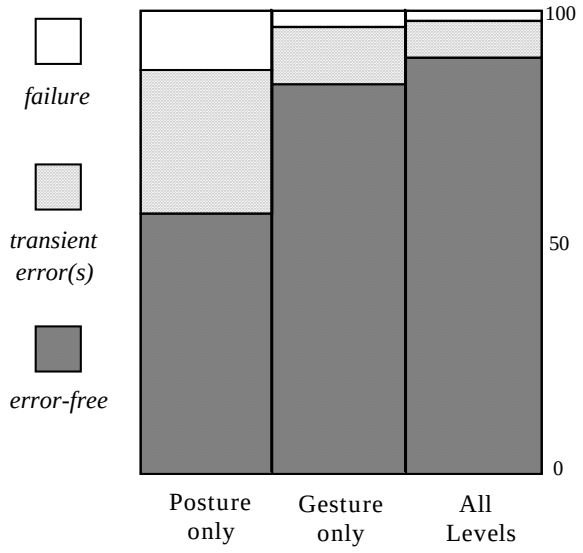

Figure 7 Average action recognition rates (%)

We distinguish three recognition possibilities: successful recognition, transient errors and recognition failure. A transient error is a recognition error that occurs during the transition intervals in the stand-perform-stand cycle. Such transient errors happen typically for gestures whose specification is not sufficiently constrained. For example, ‘wave right hand’ and ‘pointing right hand right’ (cf. Table 2) use both the gesture primitive (right hand, right). When a pointing gesture is performed, a ‘wave’ gesture might be temporarily recognised. A recognition failure is typically due to a posture recognition failure, either because of the selectivity parameter (Equation 1) or because of the algorithm of distance computation itself.

Action Response

Providing a sophisticated behaviour to synthetic actors is useless, if it cannot be expressed in a visual or audible way. In our testbeds the synthetic agents use mainly gesture responses. Motion generation for humanoid actors is a difficult task (Perlin et al., 1996). Even the smallest problems raise the attention of the observer because we all are experts, due to our every day experiences. We investigated on a framework (Boulic et al., 1997) dedicated exclusively to the integration and mixing of human motion generators. We can choose among several motion generators such as procedural walking, inverse kinematics, keyframe sequences, grasping, gazing, and specialised motions. The software architecture, called AGENTlib, handles motion blending and motion parallelism with a mechanism of weights and priorities:

- Parallelism happens when two or more motions update the body configuration simultaneously. If the involved actions use mutually non-intersecting sets of body parts, parallel actions can be performed as is. In case of intersections, a weight and priority mechanism arbitrates among the motion generators.
- Blending occurs whenever motions are performed sequentially and overlap on some period of time. The overlapping is desirable as it provides a fluid and continuous motion.

Any motion generator produces a set of joint angle values at each timestep. We apply the following equation to each DOF:

$$\theta^{k+1} = \theta^k + \sum_{i=1}^{I} f(\theta_i^k - \theta^k) \cdot \omega_i^k$$

Equation 2 Motion blending equation for one DOF with I contributing actions

where k is a time index of the associated variable

θ is the current value of the DOF,

θ_i is the DOF value of the i^{th} contributing motion generator

ω_i is the weight corresponding to θ_i. The weight follows a cubic step function, sometimes called 'ease-in ease-out', whose range is within [0.0, 1.0].

f() is a function to constrain the angle delta values; currently we use the identity function. f(x) = x

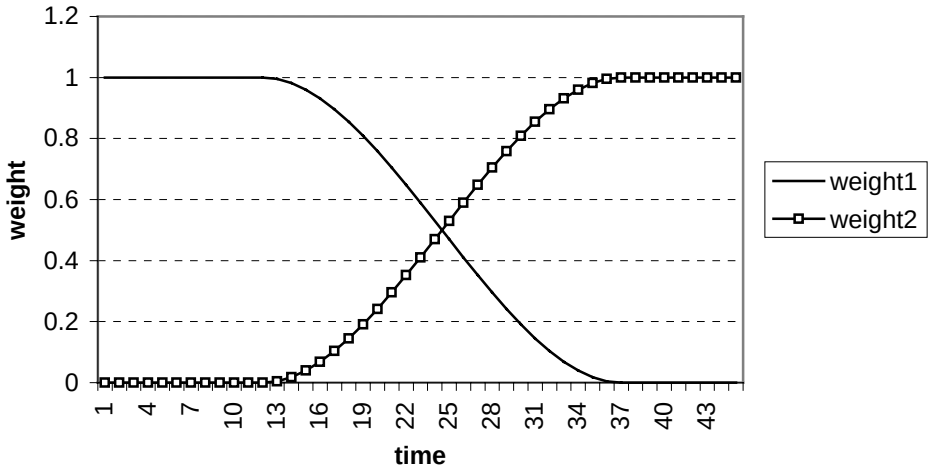

Figure 8 Illustration of Equation 2; DOF weight evolutions during the blending of two actions

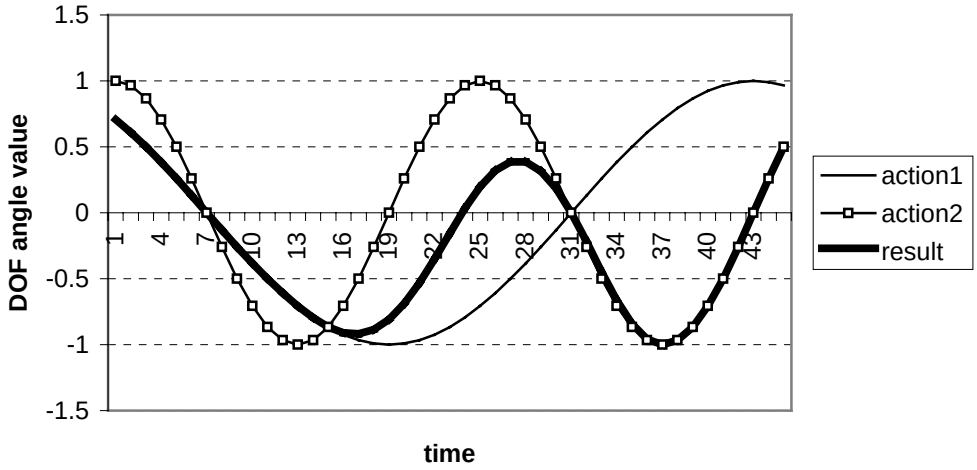

Figure 9 Illustration of Equation 2: an initial action is blended with a second one

For the global positioning and orientation, we need to blend 4x4 homogenous matrices. First we compute the delta transformations between the current matrix and the matrix of the i^{th} contributing motion generator (cf. Figure 10).

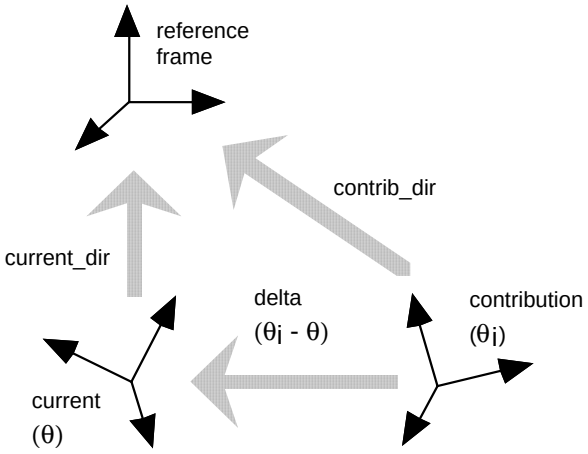

Figure 10 Motion blending with 4x4 homogenous matrix

The matrix is converted to an Eulerian vector, in other words, a vector representing the axis and whose norm is the rotation angle around this axis. A special algorithm handles the angle discontinuity at the boundaries [0, PI]. The resulting angle is multiplied by the current weight

and converted back to a matrix. All contributing matrices are multiplied together to produce the final location matrix.

Examples

In the first example, we use action recognition as a means of controlling and guiding a synthetic humanoid in a virtual environment. Figure 11 shows the subjective view of the agent.

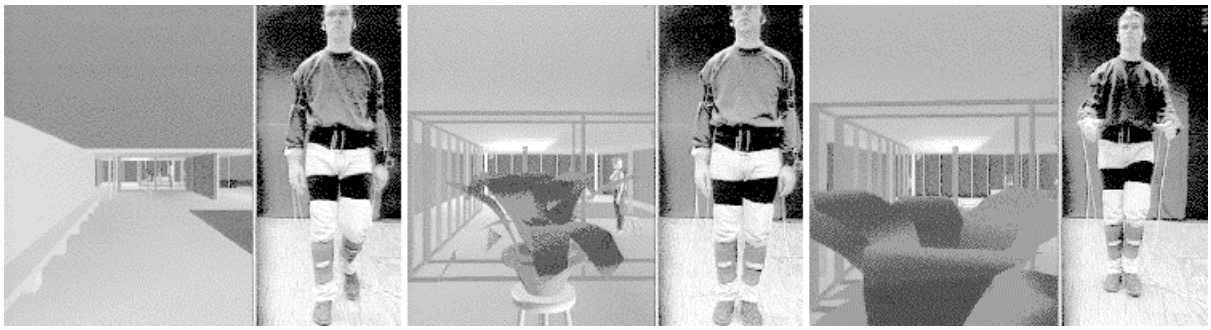

Figure 11 Interactive walk-through environment driven by action recognition events: the performer approaches the green plant, grabs it and continues to walk.

We underline that the synthetic agent is not an avatar here: rather than copying the performer's motions, it takes advantage of action recognition skills. Based on the identification results and a finite state automaton, it activates different motion generators. Table 4 summarises the actions that the agent is able to recognise and the associated responses. Some of the reactions are context-dependent: the recognition of one action activates different response schemes that depend on the current state and location of the agent.

Performer's action	Event interpretation
walking forwards	walking forwards
walking backwards	stop walking
while walking point right arm right	turn right or walk straight if currently turning left
while walking, point left arm left	turn left or walk straight if currently turning right
grasping	if agent is close to: - the object: attach or detach it - the street lamp: switch on/off global light sources
vertical head movement	tilt head vertically to inspect environment
horizontal head movement	turning the head horizontally to inspect environment

Table 4 Summary of action-response correspondences for Example 1

Although the application worked as we intended it to, this example was very revealing about pitfalls in designing good human-computer interfaces. We discuss our experiences further on in this paper.

In the second example (cf. Figure 12 & Figure 13), we use action recognition as a new skill for autonomous agents. It illustrates in a pedagogical way that, even in virtual reality, maintaining good relationships requires some personal effort. The performer is immersed into a virtual shop environment where he meets a friend named 'Mary'. The performer is supposed to buy a gift for Mary and to experience her reaction. Both the shopkeeper and Mary are autonomous agents. They use the recognition system in order to analyse the performer actions and react in consequence. When the performer points to the desired object, the shopkeeper selects it and waits for confirmation. By nodding the head, the performer can either agree or refuse if he has changed his mind. On confirmation, the shopkeeper lifts the object and waits until the participant performs a grasping action. Then the participant can hand the object to Mary. If she is happy, she responds friendly to the performer's gift and grasps the object. Finally, she accepts to go for a walk together. In the opposite case, she adopts a spiteful manners and leaves the scene. Table 5 gives a brief overview of the key events driving the automata of the example.

Figure 12 The virtual Shop. The performer tries to please a friend and naively thinks that giving a green plant will do the job (cf. MPEG files at http://ligwww.epfl.ch/~emering/AAI_JOURNAL).

Performer's action	**Event interpretation**
walking forwards	let the avatar start walking
walking backwards	let the avatar stop walking
right arm pointing forwards left arm pointing forwards right arm pointing right left arm pointing left	select one of the four objects on the table of the shop. We used gesture-based actions for illustration purposes but posture based actions would have been fine too.
grasping	pick up the selected object
say 'no'	cancel an object selection
say 'yes'	confirm an object selection ask Mary to accept the gift
wave right hand 'hello'	greet Mary; if she is within the field of view of the performer and vice versa, she responds by waving her hand and joins the performer in the shop
standing still	neutral action, it is used for synchronising the agents' automata but does not imply a behavioural response
handkiss	although recognisable and planned in the scenario, this action was not used because it was too difficult to perform the action due the limited view with the head-mounted display (HMD)

Table 5 Summary of action-response correspondences for the Virtual Shop.

The motions of the autonomous agents are performed with AGENTlib. Essentially, they use play back of pre-recorded keyframe sequences, inverse kinematics, procedural walking and look-at motions.

The main advantage of Virtual Reality compared to real life is that a scenario can be repeated. Figure 13 shows another take of the Virtual Shop environment. Note that the performer is not wearing a HMD this time. Instead we use two distinct displays, adequately arranged for the performer.

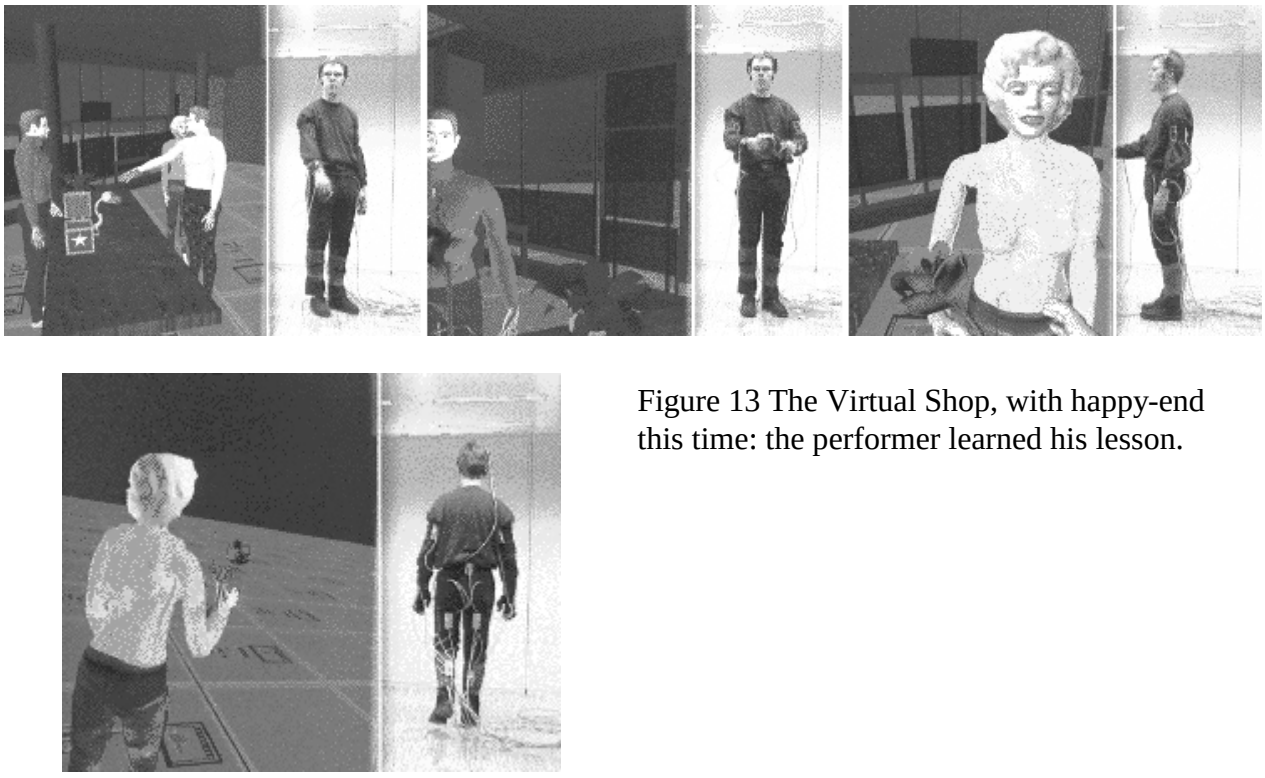

Figure 13 The Virtual Shop, with happy-end this time: the performer learned his lesson.

Discussion

In this section we discuss our experiences with human action recognition as a bilateral communication channel between a performer and life-like synthetic agents. First we discuss

some user experiences in relation with the illustrations previously presented. These are followed by some thoughts about acceptance of full-body recognition as a human-computer interface and some future work.

User Experiences

Here we want to share the experience gathered during the performing of the examples described in the previous section. Most of the issues discussed here are drawn from the action guided walk-through illustration, but they apply to VR testbeds in general.

- event / action associations. The action recognition system generates events which are associated with actions (cf. Table 4 & Table 5). Those associations have to be chosen with great care as the convenience of the human-computer interface heavily depends on them. Adjusting the displacement direction by raising the left or right arm is an example of a bad association. If our first goal was to illustrate the action recognition system, it would be a bad choice to use arm-based motion control for an interactive walk-through environment. A direction control connected directly to the relative orientation of the performer is much more intuitive and efficient. Choosing the right event/action association is a crucial aspect in a human-computer interface. However, there are some examples of associations that cannot be performed in a natural way. 'Walking' is such an example. The recognition system allows identifying a 'walking' action. Nevertheless, technical limitations make it impossible for a performer to just 'walk' through an environment. Some researchers experienced with 'treadmills' (Pratt et al., 1997), but treadmills don't allow an easy solution for the walking direction. Our solution consists of splitting the event into two parts: when the performer performs a few steps forwards, we activate the 'walking' motion generator and keep it performing even when the performer stops walking. The 'walking' motion generator is de-

activated when the performer walks back a few steps. This approach is a compromise between the limited space available for the performer's activity and the need for large range displacements in a virtual world. The walk forward and backward mechanism works fine as long as it approximately corresponds to an identity operation, i.e. globally the performer does not move away from the original position. In general however, this condition is not satisfied. When the displacements are accompanied by radical orientation adjustments, the performer will inevitably drift away from the allocated activity space.

- realistic behaviour. If we create life-like environments and inhabitants, we should expect life-like behaviour from the performer. We use a 'grasp' action recognition event to attach a near-by object to the performers' hands (cf. the plant in Figure 11 and the rose in Figure 13). To continue our quest for realism, the object should at least have the property of being fixed to a support. As long as the performer keeps a grasp-like posture, we can reasonably suppose that the performer's hands give the object's support points. If the performer changes his posture, this supposition no longer holds: the object should drop to the floor due to elementary gravity properties. This example was actually the main motivation for using constraints.

- need of Head Mounted Display (HMD). Developing 3D interactive environments with full body involvement of a performer requires a 3D activity space for the participant. A 3D activity space needs non-localised visualisation possibilities. Currently, two solutions are explored in the VR community: a CAVE system or the use of head mounted displays. For our testbeds we used the second alternative. Although a HMD allows a true 3D stereo view, we found its use somehow restrictive for high quality immersion. For example, a realistic depth feeling could not be achieved. This is quite embarrassing, as distance estimations are essential

for walk-through environments. For example, for grasping an object, we reasonably consider that the performer's distance to the object should be smaller than the length of the arm. In the example of Figure 11, we found out that this is hard to achieve. Another aspect is the vision field of vision of the HMD. In real life, we focus on one part of the scene, but we are aware of a much larger field of vision. For example, we see our hands although we don't focus on them. With an HMD, you lose awareness of your body, you only see the region of focus. When the performer is close enough to the plant object in order to grasp it, a large percentage of his view is filled up with the plant, itself thus hiding the scene. This missing lateral view makes it difficult to know where you are walking or what you are grasping. It forces you to explicitly check your body location with respect to the floor or the location of your hands with respect to yourself and the object. Under these conditions, precise navigation or environmental interactions are difficult. In our 'walk-through' example, we experienced this phenomena by a frequent motion direction adjustment by the performer: it is hard to steer straight forward in direction of a target.

- time. When immersed into virtual environments, the feeling of time is changed. As one is too busy contemplating the environment in order to localise oneself in the scene, one spends much more time on a task than is needed in real life. This is especially true for VR immersions through a head-mounted display.

User Acceptance

Here we discuss a few relevant aspects that we found to be important for the user's acceptance of a human-computer interface:

- setup. The setup time of an interactive application is a major issue in interfaces. Standard interface devices such as mouse, keyboard or spaceball take a non-perceivable setup time for the user. In VR applications, the performer's state has to be measured. This can only be done with a more or less complex sensor system that has to be attached on to the performer. All our VR testbeds have been performed with a magnetic motion capture system. This means that a dozen sensors have to be fixed to the performer besides the setup of the HMD and the datagloves. Currently this procedure is too heavy for frequent trips into cyberspace. Nevertheless wireless technology is emerging and will eventually reduce the hardware setup time to a negligible amount. On the software side, our action recognition system is based on generic body information. Its setup time consists of explaining to the user the type of actions that can be recognised.

- feedback. Independently of the form used, be it visual, audio or force, feedback is of major importance in interfaces. In our case the feedback of the interaction consists of behavioural responses from the synthetic agents. Currently we use only non-verbal communication: the agents react by some gestures to the user's actions. In a more complete testbed, the response would be extended to some speech and facial expression response. The behavioural complexity of agents depends solely on the imagination of the programmer as long as the necessary environmental perception tools are available for the synthetic agent. We argue that action recognition is one of these tools.

- user sensitivity to details. Finally we mention that humans are very sensitive to the interface with a computer, as well as to the synthetic environment they interact with. Most often, small details add a lot to the user acceptance of a system. From a graphical point of view, this might simply be the presence of shadows of the agents' embodiments or their

moving chest due to breathing simulation. For the interface, users will more readily accept a misinterpretation than the ignorance of their input. The problem is that we all are experts when we are confronted with life-like agents: we experience it every day. Therefore we are very critical of small errors, be they motion, visual or response errors.

Future work

- posture algorithm. As action recognition is supposed to be a skill among others for synthetic agents, the algorithms have to perform much faster than real-time. Our hierarchical model helps balancing algorithmic complexity and recognition speed. Nevertheless, we plan to replace the rather simple distance selection algorithms (cf. Equation 1) by a more sophisticated approach based on radial basis function (RBF) neural networks. In this context, an RBF network can be compared to an activity surface in posture space. Close to a posture prototype, the surface is at the maximum activity value. The activity decreases non-linearly with the distance of the posture from the prototypes. A major expected improvement is a higher resolution in posture selection. This allows the defining of a larger database of postures and the increasing of tolerance of posture similarity. Another interesting property of RBF networks is the possibility of defining several prototypes for a same posture. The idea is to have different people perform a given posture and store all these prototypes in order to define the posture. The RBF produces a smoothly blended activity shape around these prototypes in the activity map. It is comparable to a continuous bounding volume in posture space. The posture is recognised if it is within this volume.

- gesture algorithm. Currently, gesture primitives only allow the recognition of simple motions, such as walking, pointing and grasping. Precise motions at the end effectors, for example, cannot be recognised. The problem lies within the combination of gesture

primitives for gesture specifications. At this time, a gesture is defined by a single Boolean expression of primitives. In order to increase the resolution, we need to extend the temporal dimension of gestures. We need to keep trace of the activation of action primitives, i.e. an activation history. Then a gesture is defined by a sequence of gesture primitive expressions. Another improvement for gestures are cinematic aspects. The system is able to recognise a walking gesture, but we do not know anything about the speed. Recognising acceleration parameters in gestures is important as they can modulate the meaning of a gesture itself.

- skeleton DOF weighting. Our current virtual body model has roughly 68 DOF, excluding hands and feet. Obviously, all do not have the same importance during a gesture or a posture: there are always some irrelevant DOF to the action. Thus using all 68 DOF for action definition leads to over-specification. Reducing the number of joints to the ones relevant in the action decreases the sensitivity to noise. One solution is to associate a weight to each DOF. The weight values 0.0 and 1.0 correspond to minimal and maximal DOF relevance. Due to the number of DOF, manual weight adjusting has to be excluded. Doing it automatically requires several samples of the same action. The DOF variations in the samples carry information about the DOF influence: the larger its variations, the less relevant is the DOF. On the other hand, the smaller the variations, the higher its relevance. Using weights for each DOF allows the reducing of the number of joints while increasing the specification precision and therefore the recognition performance. Finally, we would like to mention that our current anatomical converter (Molet et al., 1996) does not necessarily animate all 68 DOF of the avatar. It depends on the number of sensors effectively used and the selected conversion algorithm. The weight mechanism would also filter out the unused joints. Note that we currently have the possibility of specifying a set of relevant body parts per action.

This is actually a special case of DOF weighting already: all DOF of the used body parts are assigned the maximum weight, and all other DOF are assigned a weight zero.

- body part extensions. So far, we did not use the hands in illustrations of our recognition system although they are as integrated as any other DOF of the virtual body. The reason for this is a bad correspondence between the virtual hands and the performer's hands when using datagloves. An automatic calibration is needed prior to further investigation in this direction. By analogy to our body data decomposition, we plan to use three data levels for the hands: hand centre, finger tips, and finger joints.

Conclusion

In this article, we presented action recognition as a skill for establishing bilateral interaction between human performers and autonomous agents. We proposed a possible decomposition of human activity into actions that can be identified in real-time by a computer system. We also covered the response to actions in the context of a motion-mixing framework. Two illustrations of a performer interacting with life-like agents enabled us to collect some experiences and to propose further enhancements. Our conclusion is that the presented action model and recognition system has a promising potential in cases where life-like interaction is needed.

Acknowledgements

The authors would like to thank S. Rezzonico for the sensor software interface, T. Molet for his real-time anatomical converter, J. Shen and E. Chauvineau for the human body deformations and all the members of EPFL-LIG for their constructive help for the present work. The research was partly supported by the Swiss National Foundation for Scientific Research and the Federal Office for Education and Science in the framework of the eRENA Esprit LTR project.

References

Amaya K., A. Bruderlin, T. Calvert. 1996. *Emotion from Motion*. In Proc. Graphics Interface 1996, pp. 222-229, 22-24 May, Toronto, Ontario, Canadian Human-Computer Communications Society, ISBN 0-9695338-5-3

Bobick A., S. Intille, J. Davis, F. Baird, C. Pinhanez, L. Campbell, Y. Ivanov, A. Schütte, A. Wilson, 1997. *The KidsRoom: A Perceptually-Based Interactive and Immersive Story Environment.* M.I.T. Media Laboratory Perceptual Computing Section Technical Report No. 398, http://bobick.www.media.mit.edu/cgi-bin/tr_pagemaker

Bobick A., and Davis J.W., 1997. *Action Recognition Using Temporal Templates* In M. Shah and R. Jain (Eds.) Motion-Based Recognition, Computational Imaging and Vision, Kluwer Academic Publishers, pp 125-146, ISBN 0-7923-4618-1

Boulic R., P. Bécheiraz, L. Emering, D. Thalmann. 1997. *Integration of Motion Control Techniques for Virtual Human and Avatar Real-Time Animation.* In Proc. VRST'97, pp. 111-118, September 1997, ACM Press.

Boulic R., S. Rezzonico, D. Thalmann, 1996. *Multi-Finger Manipulation of Virtual Objects*. In Proc. ACM Symposium on Virtual Reality Software and Technology VRST'96, ISBN 0-89791-825-8, pp 67-74, Hong-Kong, July 1996

Cassell J., C. Pelachaud, N. Badler, M. Steedman, B. Achorn, T. Becket, B. Douville, S. Prevost, M. Stone, 1994. *ANIMATED CONVERSATION: Rule-based Generation of Facial Expression, Gesture & Spoken Intonation for Multiple Conversational Agents.* In Proc. SIGGRAPH'94, pp 413-420

Darrell T., P.Maes, B. Blumberg, A. Pentland, 1994. *A Novel Environment for Situated Vision and Behavior,* Proc. of the IAPR Workshop on Visual Behaviours, pp 68-72, IEEE Press

Emering L., R. Boulic, S. Balcisoy, D. Thalmann, 1997. *Real-Time Interactions with Virtual Agents Driven by Human Action Identification*, First ACM Conf. on Autonomous Agents'97, pp. 476-477, Los Angeles - Marina Del Rey

Foner N. Leonard. 1997. *Entertaining Agents: A Sociological Study*, proceedings of *Autonomous Agents '97* conf., Marina del Rey, California USA, pp 122-129

Gavrila D.M., L.S. Davis, 1995. *Towards 3D Model-Based Tracking and Recognition of Human Movement.* Proc. of Int. Workshop on Face and Gesture Recognition, Zurich, Switzerland, 1995

Gavrila, L.S. Davis, 1996. *3D Model-Based Tracking of Humans in Action: A Multi-View Approach*. Proc. of IEEE Conf. on Computer Vision and Pattern Recognition, pp 73-80, San Francisco, USA, June 1996

Hodgins, J. K., Wooten, W. L., Brogan, D. C., O'Brien, J. F.. 1995. *Animating Human Athletics*. In Proc. SIGGRAPH '95

Kroemer K.H.E., H.J. Kroemer, K.E. Kroemer-Elbert. 1990. *Engineering Physiology, Bases of Human Factors/Ergonomics*. 2nd ed., Van Nostrand Reinhold Book, ISBN 0-442-00354-4

Looney Carl G., 1997. *Pattern Recognition Using Neural Networks*. Oxford University Press 1997, ISBN 0-19-507920-5

Maes P., T. Darrell, B. Blumberg, A. Pentland, 1995. *The ALIVE System: Full-body Interaction with Autonomous Agents.* In Proc. Computer Animation'95, pp 11-18, Geneva, Switzerland, IEEE Computer Society Press, Los Alamitos, California, ISBN 0-8186-7062-2

Molet T., R. Boulic, D. Thalmann, 1996. *A Real-Time Anatomical Converter for Human Motion Capture*. 7th EUROGRAPHICS Int. Workshop on Computer Animation and Simulation'96, pp. 79-94, Poitier, France, G. Hegron and R. Boulic eds., Springer-Verlag Wien ISBN 3-211-828-850

Molet T., R. Boulic, D. Thalmann, 1998. *Human Motion Capture Driven by Orientation Measurement*. To appear in Presence, Vol. 8, MIT Press

Newby G., 1994. *Gesture Recognition Based Upon Statistical Similarity*. Presence 3(3), pp 234-243, MIT Press

Pausch R., J. Snoddy, R. Taylor, S. Watson, E. Haseltine. 1996. *Disney's Aladdin: First Steps Towards Storytelling in Virtual Reality*. In Proc. SIGGRAPH'96, pp 193-203

Perlin K., A. Goldberg. 1996. *Improv: A System for Scripting Interactive Actors in Virtual Worlds*. In Proc. SIGGRAPH'96, pp 205-216

Pratt D. R., S. M. Pratt, P. T. Barham, R. E. Barker, M. S. Waldrop, J. F. Ehlert and Ch. A. Chrislip, 1997. *Humans in Large-Scale, Networked Virtual Environments.* Presence 6(3), pp 547-564, MIT Press

Rezzonico S., R. Boulic, Z. Huang, N. Magnenat-Thalmann, D. Thalmann, 1995. *Consistent Grasping in Virtual Environment based on the Interactive Grasping Automata, in Virtual Environment*. Martin Goebel Editor, ISBN 3-211-82737-4, pp 107-118, Springer-Verlag Wien, October 1995.

Rickel J., Lewis Johnson W., 1998. *Animated Agents for Procedural Training in Virtual Reality: Perception, Cognition, and Motor Control*. This volume.

Rubine D., 1991. *Specifying Gesture by Example*. In Proc. SIGGRAPH'91, pp. 329-337

Roy D. M., M. Panayi, R. Foulds, R. Erenshteyn, W.S. Harwin , R. Fawcus, 1994. *The Enhancement of Interaction for People with Severe Speech and Physical Impairment through the Computer Recognition of Gesture and Manipulation*. Presence 3(3), pp 227-235, MIT

Semwal S. K., R. Hightower, S. Stansfield, 1996. *Closed Form and Geometric Algorithms for Real-Time Control of an Avatar.* In Proc. of IEEE VRAIS'96, pp. 177-184, IEEE Press

Yanghee N., Wohn K.Y., 1996. *Recognition of Space-Time Hand-Gestures using Hidden Markov Models.* In Proc. VRST'96, pp 51-58