Virtual Reality & Programming by Demonstration: Teaching a Robot to Grasp a Dynamic Object by the Generalisation of Human Demonstrations.

Ludovic Hamon, Philippe Lucidarme, Paul Richard, Emmanuelle Richard
LISA Laboratory
University of Angers
62, Avenue Notre Dame Du Lac
49000 Angers, France
E-mail: ludovic.hamon@etud.univ-angers.fr, paul.richard@univ-angers.fr

Humans possess the ability to perform complex manipulations without the need to consciously perceive detailed motion plans. When a large number of trials and tests are required for techniques such as learning by imitation and programming by demonstration, the virtual reality approach provides an effective method. Indeed, virtual environments can be built economically and quickly and can be automatically re-initialised. In the fields of robotics and virtual reality, this has now become commonplace. Rather than imitating human actions, our focus is to develop an intuitive and interactive method based on user demonstrations to create human-like, autonomous behaviour for a virtual character or robot. Initially, a virtual character is built *via* real-time virtual simulation in which the user demonstrates the task by controlling the virtual agent. The necessary data (position, speed etc.) to accomplish the task are acquired in a Cartesian space during the demonstration session. These data are then generalised off-line by using a neural network with a back-propagation algorithm. The objective is to model a function that represents the studied task, and by so doing, to adapt the agent to deal with new cases. In this study, the virtual agent is a six-degrees-of-freedom arm manipulator, Kuka Kr6, and the task is grasp a ball thrown into its workspace. Our approach is to find the minimum number of necessary demonstrations while maintaining an adequate task efficiency. Moreover, the relationship between the number of dimensions of the estimated function and, the number of human trials is studied depending on the evolution of the learning system.

Nowadays, robots have replaced humans in performing a wide range of tasks. They have been used to increase productivity and efficiency through the automation of manufacturing processes. Increasingly important in recent years is the possibility of integrating robots into our everyday lives. Robots are part of our environment not only because they can perform simple and repetitive tasks, but also because they can perceive humans and a human environment. More and more robots possess learning skills whereby they can evolve. Consequently, learning by imitation, modelling human motion and behaviour have become common fields for many research areas such as robotics (Kuniyoshi, Inaba, & Inoue, 1994), cognitive science (Brown & Burton, 1978), or user-modelling (Webb, Pazzani, & Billsus, 2001).

A simple task can be performed more easily by a human than a robot, such as moving an arm, because humans possess the ability to perform complex manipulations without the need to consciously perceive detailed motion plans (Asada & Izumi, 1989). Virtual Environments (VEs) can be built economically and quickly and can be automatically re-initialised. Therefore, when a large number of trials and tests are required, for techniques such as learning by imitation (Kawasaki, Furukawa, Ueki, & Mouri, 2008 ; Chen & Naghdy, 2002), Virtual Environments (VEs) provide an effective method. As far as human perception ability is concerned, programming by demonstration (PbD) in VEs for robots and virtual characters has become a relevant domain of study, in which interactive methods have been developed to imitate human actions (Aleotti & Caselli, 2010). This technique is particularly useful for tedious, unintuitive and time-consuming aspects of robot programming (Chong, Ong, Nee, & Youcef-Youmi, June 2009).

Rather than simply imitating human demonstrations, our interest lies in the ability to provide an agent with human-like, autonomous behaviour, while maintaining an intuitive and interactive learning method. For this purpose, a particularly interesting study task in the field of robotics is grasping or catching a dynamic object. Indeed this requires dynamic, and kinematic models, as well as real-time solutions in order to, (i) detect the moving object (Riley & Atkeson, 2002), (ii) follow and predict its trajectory (Payeur, Le-Huy, & Gosselin, 1995), (iii) define and reach the meeting point (Miyazaki & Mori, 2004) and finally, (vi) grasp the objet (Aleotti & Caselli, 2010). Moreover, playing catch is a simple, safe and fun way for a person to interact with a robot, and it can be an interactive task that directly engages the user (Riley & Atkeson, 2002).

This paper describes a semi-real time, and interactive

method to create autonomous behaviour of a virtual agent *via* the generalisation of human demonstrations. In the present study, the task is to grasp a ball flying through the air, and the robot is the six-degrees-of-freedom arm manipulator Kuka Kr6. After building a VE in which the robot, and the task are simulated in Cartesian space, our approach consists of: (i) merging the user in the robot workspace with a 3D interface, (ii) demonstrating in real time the task by controlling the virtual agent and, (iii) generalising off-line data acquisition from the human demonstrations, by using a neural network and a back-propagation algorithm in order to estimate the function that represents the task and makes the agent autonomous.

In Section 1, related works about the grasp of dynamic objects in robotics, programming by demonstration, and learning through Virtual Reality (VR) simulations are presented. The virtual environment is detailed in Section 2 with a description of the Kuka Kr6 robot and the real-time control system. Elements about the chosen interface, the simulation constraints, and the different workspaces are also included in this Section. Section 3 shows the learning process, the determination of the necessary data that must be acquired and, the construction and use of the artificial neural network. The experiments and results are discussed in Section 4. An initial approach, in which the data are collected from a large number of trials to ensure the validity and the effectiveness of the method, is described, followed by an analysis of the relation between the number of trials and the number of dimensions of the estimated function. Finally, a second approach minimises the necessary number of demonstrations while keeping the same performance level for the task. Finally a discussion is provided in Section 5 followed by a conclusion and the perspectives of this work.

Contribution

A key contribution of this work is to proposed a semi-real time intuitive and interactive method to create human-like autonomous behaviour for a virtual agent or robot, based on human demonstrations.

The principal originality of the presented method lies in the dual use of the two following aspects: the first aspect is the capacity of human natural perception and planning to simplify part of the necessary and complex, dynamic and kinematic models to predict and follow a trajectory and to determine the catching point. The second one is the use of the neural network. Most the previously-cited works built a neural network or a similar system in order to classify situations or grasp recognition (Friedrich et al., 1999 ; Rose III, Sloan, & Cohen, 2001 ; Wojtara & Nonami, 2004 ; Cooper, Hertzmann, & Popovic, 2007). In this paper, the system generalises in Cartesian space the necessary data recorded from human demonstrations in order to enable an agent to accomplish the task. The function modelled by the neural network allows the robot to adapt to new cases and create agent autonomy.

It can be noted that, owing to the VR approach, a control system is implemented in which the user demonstrates the task in real time while respecting the virtual agent's workspace. Another relevant attribute of the present work is the study of the consequences linked to the variation of the dimension number of the estimated function to the necessary number of trials. Finally, we consider the approach with which to determine the minimum of number demonstrations while maintaining the same performance level of the task.

1 Related Works

1.1 Grasping Dynamic Objects

The complex robotic task consists of grasping a dynamic object. The robot usually receives the position and orientation of the object and computes the best configuration and the minimum force needed to grasp the object. This requires solving advanced kinematic and dynamic equations. In a real environment, this research field has been widely investigated. For example, Hove and Slotine built a trajectory-matching algorithm that combined an observer with a varying-strength filter, an error estimator, and an initial motion algorithm (Hove & Slotine, 1991). All path-planning for the catch occurred in real-time during the half-second that the targeted object was airborne. Rizzi and Koditschek worked on a three-degrees-of-freedom arm robot which could juggle with a ball (Rizzi & Koditschek, 1992, 1993). They also presented other interesting work such as an architecture which provided field rate estimates of the positions and velocities of two independent falling balls. Buehler and coworkers used a class of control algorithms, the 'mirror algorithms', which gave rise to experimentally observe juggling and catching behaviour in a planar, robotic mechanism (Buehler, Koditschek, & Kindlmann, 1994). Payeur *et al.* used a multilayer perceptron neural network to predict the trajectory of a dynamic object (Payeur et al., 1995). The position, velocity and acceleration of the object were predicted with past history inputs. The study was made to solve the real time trajectory problem in a robotic context wherein a manipulator had to grasp a moving object which followed an unknown path. Riley and Atkeson worked on a robot which caught balls (Riley & Atkeson, 2002). They generated ball-hand impact predictions based on the flight of the ball. According to the Programmable Pattern Generators (PPGs), and by using a stereo, colour vision system, the QuickMag, to detect and track the flying ball, the humanoid robot could move to the impact position. Two years later, Miyazaki and Mori presented an interesting trajectory control method using a visual servo and based on a motion method call GAG (Gaining Angle of Gaze) (Miyazaki & Mori, 2004). This method focused on the angle tangent of gaze against the ball and, by keeping the angle value gaining in a finite rate-of-change, they assumed that the mobile robot was able to track and catch the ball in a three dimensional space. Hirai and Miyazaki described the hierarchical architecture for rhythmic coordination among robots which suited juggling-type tasks (Hirai & Miyazaki, 2005).

These previous works are relevant, and their corresponding results show an interesting degree of performance and

realism. However the analysis and implementation of these methods added to robot programming, make these processes time-consuming and unintuitive (Chong et al., June 2009).

1.2 Programming by Demonstration

Grasping dynamic objects requires real time solutions, processes and models to represent human action. With a robotic arm manipulator, it may become an interactive task which is in relation with human behaviour and motion. Programming by Demonstration (PbD) primarily addresses intuitiveness in human-robot interaction. With this approach, robots can learn, improve or duplicate actions based on human demonstrations which provide an intuitive and fast way of programming. PbD-systems became a common way of tutoring and programming robots for trajectory oriented tasks, such as path planning through obstacles (Delson & West, 1994), or for industrial tasks, such as arc welding or gluing (Schraft & Meyer, 2006). Different PbD approaches have been proposed and reviewed (Dillmann, Rogalla, Ehrenmann, Zollner, & Bordegoni, 1999 ; Knoop, Pardowitz, & Dillmann, 2008). In some cases, these systems require a large number of demonstrations before the learning process can start. However, according to Kawasaki *et al.*, in the robotic field, learning by imitation is more effective in a VE. Indeed, the VE can be built economically and can be quickly re-initialized (Kawasaki et al., 2008).

1.3 The Virtual Reality Approach

During a learning session, the following drawbacks can appear when a robot is manipulated in a real environment (Tsubone, Kurimoto, Sugiyama, & Wada, 2008): (1) the large number of tests needed to acquire adequate actions may exceed several thousand, hence the restoration of the environment to the initial learning state can be tedious, (2) in the learning process there are safety problems with the robot itself, as well as the possibility of damage to operational objects, robotic arms and/or human bodies, if the robot's movements are not stable, (3) the operator undergoes continued stress because a mistake will immediately be copied by the robot and may result in some possible damage.

VEs can overcome previous problems and offer a number of advantages including (Chen & Naghdy, 2002): (i) the training data can be extracted and recorded directly leading to the simplification of the data collection process, (ii) the environment can be modified easily if the manipulation process and its requirements are changed, (iii) the risk of breakdown and system breakage is very low, (iv) dangerous environments can be built and simulated and, (v) a user-friendly environment for the human operator can be developed easily.

VEs have often been used in order to study and imitate human action and behaviour. For example, Kawasaki *et al.* developed a system of virtual teaching for multi-fingered robots (Kawasaki, Nakayama, Mouri, & Ito, 2001). Experimental results of a pick-and-place task were presented to demonstrate the effectiveness of the proposed method in a VE. Chen *et al.* studied the transfer of manipulation skills from a human to a robotic manipulator through a task demonstration in a haptic-rendered VE (Yutuo, Xuli, Okada, Chen, & Naghdy, 2007). They taught a 'peg in hole' insertion task to a robot, and studied how human manipulation skills could be replicated. Knoop *et al.* presented an approach for mapping symbolic task knowledge that had been obtained from user demonstrations by a PbD system to a specific robot (Service robot Albert 2) (Knoop et al., 2008). Chong *et al.* reviewed some VR systems used for object manipulation in environments that were known *a priori* (Chong et al., June 2009). Aleotti *et al.* exploited the automatic reconstruction of a VE (Aleotti & Caselli, 2010). Indeed, based on classic patterns in Augmented Reality, their system first built a VE matching the real environment in order to easily configure object placement. The user had to demonstrate a number of grasps performed on static trial objects. Their programs learned and classified human grasps off-line, and the demonstrations were then imitated and adapted to a specific robot device.

1.4 The Active Learning Approach

In the domain of motion capture, a recurring problem is that of building compact controllers for virtual, human-like agents from a vast amount of motion samples and data. Recently, Cooper and co-workers reviewed this field and developed a method to obtain compact, motion controllers for complex parameterised tasks (Cooper et al., 2007). They developed a real-time active learning system with the user as a key-point of the process. Their method consisted of using existing data of human motions and a task to perform (for example, catching a ball), and identifying cases when the virtual agent performs poorly. The system then creates new pseudo-examples of motions by concatenating and interpolating existing ones. If there is a sample considered as acceptable by the user, it is recorded, otherwise, the user performs the task during a motion-capture session and the result is recorded. Numerous end criteria of the process are defined, one being when all 'poor-performing' candidates appear to be high-quality to the user. A catching task was built with a task success rate of 57 %, notably obtained by 30 acquired samples. In terms of task efficiency, this result seems to be insufficient and can be in relation to the end criteria implicating the user judgment. However, the purpose of this research was not based on task performance but in the way to quickly and interactively build highly-compact motion controllers.

2 Application Design

2.1 Description

This application is written in C/C++ language with the OpenGL library. The simulation contains virtual objects such as a ball, a hand controlled by the user and a robotic arm manipulator Kuka Kr6 (Figure 1). The virtual ball is animated according to Newton's second law: $\sum \vec{F} = m\vec{a}$ ($\vec{F}$: force vector, m: masse, $\vec{a}$: acceleration vector).

2.2 Kuka Kr6 Specifications

The Kuka Kr6 robot is an arm manipulator with 6 axes of rotation and joint-arm kinematics. Figure 2 shows its

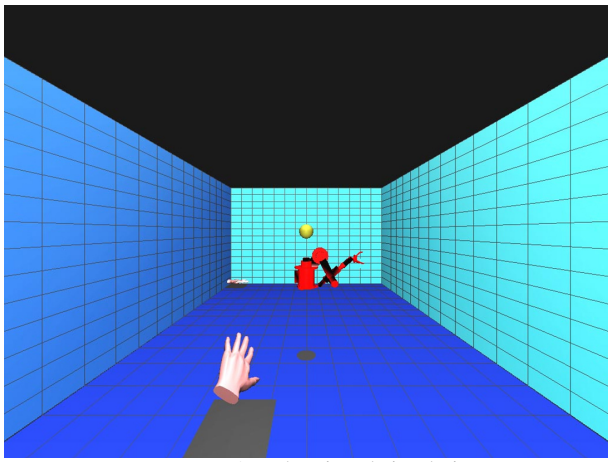

Figure 1. The virtual simulation.

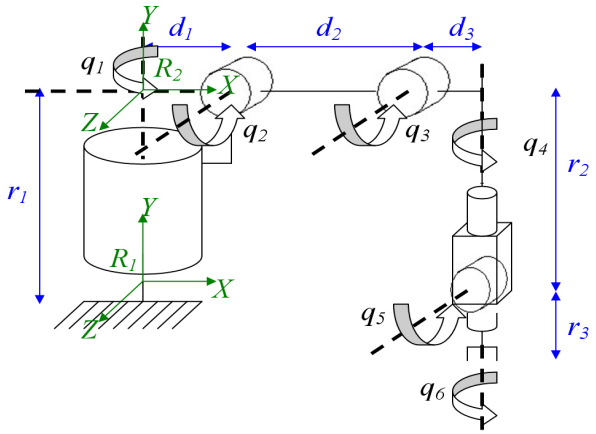

Figure 2. Parameters used for the direct geometrical model of the Kuka Kr6.

topology. The gripper has a Cartesian position according to the six angles q_1, q_2, q_3, q_4, q_5, q_6 given by the direct geometrical model defined by X=f(Q), with X=(x,y,z) and Q=(q_1,q_2,q_3,q_4,q_5,q_6) (cf. the Denavit-Hartenberg parameters, (Spong & Vidyasagar, 1989), Table 1).

To get the values of the six rotation angles according to the gripper position, the inverse geometrical model based on Paul's method is used (Paul, 1979). With the hypothesis that the robot can reach all points of its workspace at any time, with no constraints on the rotation angles, the Kuka Kr6 workspace is a torus defined by the following equation:

$$(\sqrt{(x^2 + z^2)} - M)^2 + y^2 \leq R^2$$

with:
$-R = \sqrt{((r_2 + r_3)^2 + d_3^2)} + d_2$
$-M = d_1$

Table 1
Denavit-Hartenberg Table.

j	a	d	q	r
1	0	0	q_1	r_1
2	90	d_1	q_2	0
3	0	d_2	q_3	0
4	90	d_3	q_4	r_2
5	-90	0	q_5	0
6	90	0	q_6	r_3

Four parameters are used to pass from a system of coordinates R_{j-1} to a system of coordinates R_j:
- a_j rotation angle between axis y_{j-1} and y_j around x_{j-1} axis,
- d_j distance between axis y_{j-1} and $y_j$measured along x_{j-1} axis,
- q_j rotation angle between axis x_{j-1} and x_j around y_j axis,
- r_j distance between axis x_{j-1} and x_j measured along y_j axis.

2.3 The Tracking Interface & the Workspace Correlation

Using a virtual agent like a puppet in real time leads to a workspace problem. Indeed, the implemented virtual simulation must deal with the workspace of (i) the user, (ii) the interface, and (iii) the virtual agent. The workspace and the multi-modal aspect of the interfaces create a vast study field in the VR domain. In the present study, a possibility is to chose a haptic interface that constrains the user to use the robot workspace, owing to the feedback system. Furthermore, according to Richard *et al.*, the haptic feature of some interfaces can enhance user immersion (Richard et al., 1996). However, Richard and co-workers reviewed these interfaces and argued that most of them are intrusive, expensive and have a limited workspace (Richard, Chamaret, Inglese, Lucidarme, & Ferrier, 2006). The limited workspace feature can also be considered for desktop 3D interfaces such as the *PhantomOmni*^{TM} device from Sensable, although it offers a high degree of accuracy. The use of interfaces with a large workspace, such as a motion capture system or a human-scale interface such as the SPIDAR (Space Interface Device for Artificial Reality), can be considered (Chamaret, Ullah, Richard, & Naud, 2010). The present study does not require expensive, motion-capture equipment, and the virtual agent can be different from a human model. Our interest is not in the reproduction and creation of animations. The SPIDAR, as all haptic interfaces, is confronted with the resistance and weight of elements that generate the force feedback. Indeed, for action with high levels of dynamism, the string and motors can possibly limit the user performances. This last point, added to the necessary multiple human demonstrations of such a dynamic action, led us to adopt a tracking system using the *Patriot*^{TM} from Polhemus, that has a adequate workspace for this task, in order to track the user's hand with relatively little clutter.

2.4 The Grasping Interface & System

The grasp techniques in VEs with high-accuracy, hand-like interfaces have been thoroughly investigated, in particular by Aleotti and Caselli (Aleotti & Caselli, 2010). According to them, VR gloves are not designed for physical manipulation. They are in fact, dedicated devices and can easily be broken through the repeated contact and manipulation involved in physical grasping. Consequently, a grasp technique quite similar to the work of Cooper and co-workers (Cooper et al., 2007) was adopted. A simple classic glove is built with the patriot sensor and an integrated, wireless mouse button to close or open the virtual hand. The ball can be grasped by the virtual robotic arm or by the virtual hand. A detection sphere is used to represent the physical collision of the gripper (respectively the hand) and it remains invisible during the experiment. The detection sphere has a predefined size and is adjusted to the gripper (the hand) position. If the ball centre is in the detection sphere, and if the gripper (the hand) is closed, then the ball is caught. The object position is readjusted according to the robot gripper position (respectively, the hand position, Figure 3).

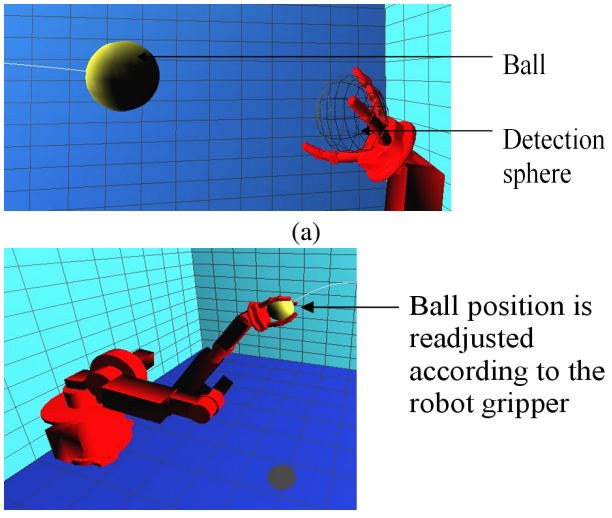

Figure 3. Grasping system: (a) the detection sphere and the flying object. (b) the ball centre is inside the detection sphere, so the dynamic object position is readjusted to the gripper position.

2.5 Robot Control System

To control the virtual robot, the user operates the Kuka Kr6 end effector with the virtual hand. The concept is to merge the virtual hand and the robot gripper (Figure 4). At any frame, the Cartesian position of the user hand is known, and if it belongs to the robot's workspace then the inverse geometrical model is used to find the six angles q_1, q_2, q_3, q_4, q_5, q_6 according to the hand position. Opengl blending effects are added on the hand and the Kuka Kr6 model. When the hand is opened, the gripper is opened and when the hand is closed, the gripper is closed. Only the robot gripper can grasp the ball. Figure 5 shows the Kuka Kr6 controlled by the user.

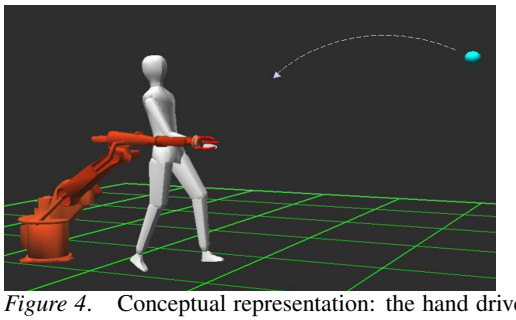

Figure 4. Conceptual representation: the hand drives the gripper to reach the thrown ball.

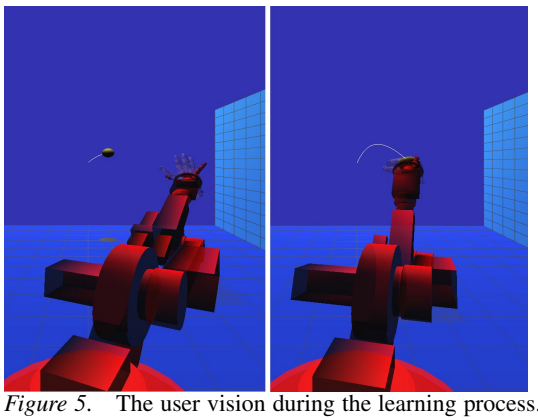

Figure 5. The user vision during the learning process. The robot is driven by user to grasp the flying ball.

Nevertheless, the kinematics of the arm robot are specific and, to adopt such a control scheme while respecting the rotation speed of each axe, for example, can make the demonstration process difficult. Consequently, as suggested in Section 2.2, no kinematic constraints on the Kuka Kr6 rotation angles were applied during the learning process.

3 Learning Process

3.1 Demonstration Process

In Cartesian space, a realistic hypothesis is that the robot can simultaneously determine the ball position (Pb_x, Pb_y, Pb_z) and its speed (Vb_x, Vb_y, Vb_z) at one or more discreet times 't1'. The gripper's position (Pr_x, Pr_y, Pr_z) therefore has to meet the flying object at a moment 't2' greater than 't1', on the ball's trajectory, within its reachable space. There are several points on the trajectory where the object can be caught. This constitutes a 'meeting problem'. To overcome the problem of defining the meeting point and, predicting and following the trajectory, the user freely drives the arm robot in order to grasp the dynamic object each time the ball is thrown.

3.2 Data Acquisition: Output & Input Determination

The constraints of the learning protocol are the following: (C1) the ball has a fixed start position, (C2) the trajectory is parabolic, (C3) the ball has to reach the robot's workspace. The process is the following one: initially, to ensure the validity of the method, a great quantity of balls are thrown until six hundred are grasped by the robot driven by the human operator. The balls are thrown with a random initial velocity vector. These vectors are the inputs of the neural network. For every grasped object, the gripper position is recorded. These vectors are the expected outputs. Knowing that six hundred speed vectors are randomly generated, the fact that there are not two identical vectors is checked. Furthermore, an average of nine hundred throws is finally necessary until the user grasps the six hundred wanted objects (a 2/3 ratio). Figure 6 shows the demonstration process.

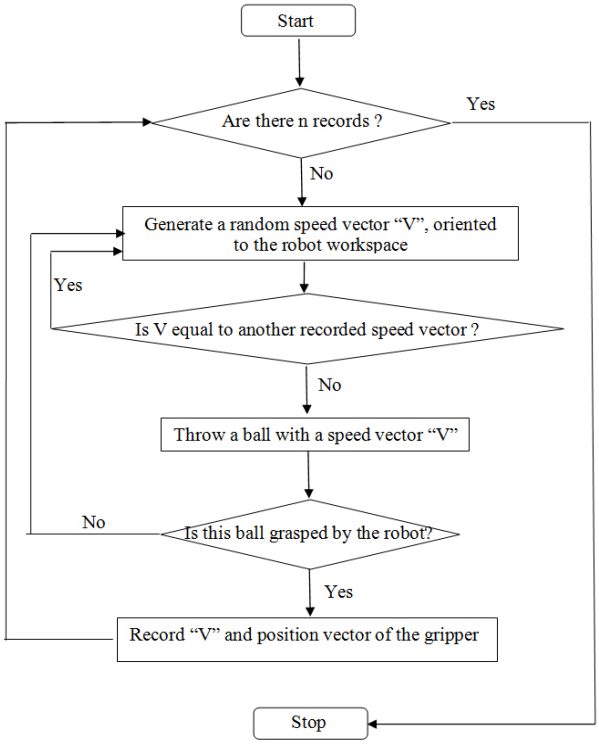

Figure 6. Diagram of the demonstration process (n: number of required demonstrations).

3.3 Artificial Neural Network

To go further than a simple imitation of human demonstrations, the autonomous behaviour of the robot is constructed from the data acquired from the human demonstrations. Indeed, it is assumed that there is a relationship between the recorded inputs and the outputs. Consequently, the objective is to find a mathematical model of this relationship knowing that the data are in closed definition domains. A model of the process based only on measured data is searched for. In an artificial neural network domain, this concept is called: 'black box modelling' (Dreyfus, Martinez, Samuelides, & Gordon, 2002). On the hypothesis that the expected solutions are probably not separated from the unwanted solutions by a hyperplane, a no-loop neural network composed of one hidden layer is used, with a back-propagation gradient algorithm (Figure 7).

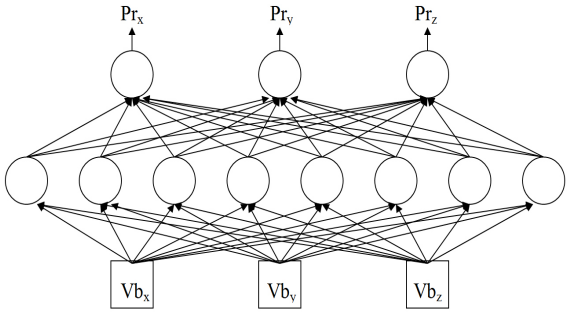

Figure 7. Neural network used for this experiment: 3 inputs, 8 neurons in 1 hidden layer, 3 output neurons, activation function: sigmoid exponential function and n = 0.2 (n: learning rate).

The following steps are executed: create a learning set, create one or several test sets, observe the iteration number (NI) of the back-propagation gradient algorithm and the quadratic error (E) on the learning set, and stop the algorithm following the values of E and NI (Dreyfus et al., 2002). To consider the global performance of the learning set (respectively test sets), the average quadratic error EQMA (respectively EQMT) is calculated:

$$E = \frac{1}{2}\sum_{e}(s_e - y_e)^2$$

$$EQMA = \sqrt{\frac{1}{Na}\sum_{ea}(s_{ea} - y_{ea})^2}$$

$$EQMT = \sqrt{\frac{1}{Nt}\sum_{et}(s_{et} - y_{et})^2}$$

- e: element belonging to the learning/test set.
- ea: element belonging to the learning set.
- et: element belonging to the test set.
- s: wanted output
- y: obtained output
- Na: number of elements of the learning set.
- Nt: number of elements of the test set.

Of the six hundred vector couples, one hundred couples compose the learning set. Then five test sets of one hundred couples each are created with the five hundred remaining couples. In the following sections, we define a *trial* as a throw belonging to a learning set.

4 Experiment & Results

In a real environment, carrying out an experiment of this sort and its re-initialisation can be tedious and complicated (cf. Section 1). The virtual reality approach allows for a quick performance process with a very large number of trials/tests in total safety. In this way, the following experiments are carried out automatically. The task coverage is defined by the percentage of caught balls with the method

Table 2
Results for the first experiment.

set of	EQMT	throw	grasp	coverage
learn.	/	100	93	93 %
test1	0.050	100	97	97 %
test2	0.067	100	95	95 %
test3	0.066	100	96	96 %
test4	0.057	100	92	92 %
test5	0.069	100	95	95 %

- E=0.049
- NI = 86000
- EQMA=0.049

Table 3
Results for the second experiment.

throw	grasp	unreachable	coverage
1000	838	82	91.3 %

Table 4
Results for the first experiment with fifty trials.

set of	EQMT	throw	grasp	coverage
learn.	/	100	93	93 %
test1	0.056	100	93	93 %
test2	0.076	100	95	95 %
test3	0.071	100	97	97 %
test4	0.061	100	94	94 %
test5	0.074	100	94	94 %

- E= 0.063
- NI= 137000
- EQMA=0.054

Table 5
Results for the second experiment with fifty trials.

throws	grasp	unreachable	coverage
1000	810	114	91,4 %

described in Section 2.4. This percentage is calculated from the number of reachable objects.

4.1 Experiment 1: Experiment with the Learning Set & the Test Sets

Each throw belonging to the learning set and the test sets is carried out. The neural network gives the position to reach by the robot gripper. Each grasp is counted and presented in Table 2 with their corresponding quadratic errors. The coverage value is greater than 90 % in each case and the results on the sets of test show that the virtual agent adapts to new cases.

4.2 Experiment 2: Experiment on Efficiency

One thousand throws are performed with a random speed vector. The number of grasped objects and the number of unreachable objects are counted. Knowing that one thousand velocity vectors are randomly generated, the fact that there are not two identical vectors is checked. The results are presented in Table 3. With a coverage higher than 90%, the results are consistent with those of the first experiment. The virtual agent has acquired an autonomous behaviour from human demonstrations with an acceptable level of efficacy.

4.3 Results Analysis & the Variation of the Dimension Number of the Estimated Function

This Section shows the hypothesis and analysis that link the number of trials and the number of dimensions of the modeled function by the neural network. To illustrate the subject, an example of dimension variation is given by an approach for raising the constraint C1 and, consequently the extension of the present study to the whole experimental space. The presented approach does not offer an optimum result and is not the only way to raise C1. However, one of the objectives of this Section is to show the consequences on the number of trials if the number of dimensions needs to be changed.

With the C1 constraint, one hundred trials are enough to generalise the process and obtain adequate results. However, the previous work is in three dimensions (speed vector (Vb_x, Vb_y, Vb_z)). The following hypothesis is therefore made: the estimated function for which the neural network is built, globally maintains the same properties by modifying the number of dimensions. It is necessary to have one hundred velocity vectors to obtain a learning generalisation in three dimensions. Thus, according to Shanon's law, for one dimension, $\sqrt[3]{100} \approx 4.64$ trials are necessary to obtain a learning generalisation. If the start position is no longer fixed, six is the new number of dimensions (speed vector (Vb_x, Vb_y, Vb_z), ball starting position (Pb_x, Pb_y, Pb_z)), thus $4.64^6 \approx 10,000$ trials are necessary to extend the study to the whole space. This number is too high, and consequently, it is necessary to decrease the current number of demonstrations. The following study with fifty throws of the previous learning set is made. The results are presented in Tables 4 and 5.

These results shows that fifty trials are enough to generalise the process. With this new value and according to Shanon's law, for one dimension $\sqrt[3]{50} \approx 3.68$ trials are necessary to obtain process generalisation and, consequently, if the starting position is no longer fixed, $3.68^6 \approx 2,500$ trials are required to extend the study following a variable starting position.

4.4 Discussion

With quite similar coverage results, fifty trials are adequate to make the agent autonomous. While maintaining an efficiency level of over 90%, the present study shows the interactive method validity to make a virtual agent autonomous

from few user demonstrations. Depending on the value of effectiveness wanted, the number of trials can still be decreased. Consequently, a first extension of this work is to find a minimum number of trials depending on a desired degree of efficiency. However, that requires a more advanced study, more tests, and it obviously depends of the chosen constraints and the expected results.

The present method allows a simplification of defining the meeting point and, a simplification of predicting and following the trajectory by using the user's perception ability. Moreover the short calculation time once the learning process has finished, the intuitive aspect of the approach, and the few necessary demonstrations are also advantageous. It can be noted that VE offers the possibility of carrying out tests with a very large number of trials in a short time. On the other hand, the requirement of a human operator, and the non follow-up of the trajectory if the ball is missed, should also be considered. Another discussion point is the random generation of trials that facilitates the generation process, but which can be confronted with a more appropriate method of generation with, for example, equally distributed vectors in the 3D space. This last point can hypothetically improve the results but depends on the simulation context.

5 Conclusion & Future Works

An intuitive and interactive method based on human demonstrations to create human-like and autonomous behaviour for a virtual character or robot has been presented in this study. By using human perception ability than can simplify some complex dynamic and kinematic studies, and a neural network that is used to generalise human demonstrations, the method consists of firstly building a computer simulation in which the virtual agent is controlled in real time by the user in order to perform the tasks in a Cartesian space. The acquired data during the demonstration process are then used by a neural network that models a function representing the task with a back-propagation algorithm.

The method was applied in a virtual simulation in which a six-degree-of-freedom manipulator has to grasp a ball thrown into its workspace. Defining the meeting point and, predicting and following the trajectory were managed by the natural perception of the user. Our approach is concentrated of searching for a minimum of trials while also maintaining an adequate level of effectiveness, and with a task coverage of over 90 %, the results demonstrate that only a few trials (50 trials) are required to obtain the adaptation of the robot to unknown cases.

An interesting point of such a method is that the time required for a learning session and the learning performance are linked to the degree of user performance during the demonstration session. This last feature is obviously in relation with the type of interface used and the correlation of different workspaces (user, virtual agent, and interface). In the present study, although the immersion enhancement with advanced haptic features has been cited, the need to have an interface adapted to such a engaging and dynamic task with a large workspace, led to choosing a simple, magnetic tracking system. This choice can be discussed and can possibly be the cause of the user performance level during the demonstration sessions with a ratio of $2/3$.

In future research, the correlation of the different kinds of workspace depending on the 3D interface chosen will be investigated in more depth in order to increase user performance during the learning session. A more realistic simulation will be implemented with the complete physicalisation of the virtual world. A more advanced study concerning the minimum number of trials depending on a fixed efficiency value will also be carried out. Finally, the present method will be applied for other dynamic tasks in relation with various domains such as industry, interaction between human and robot, and advanced human behaviour modelling.

Références

Aleotti, J., & Caselli, S. (2010). Grasp programming by demonstration in virtual reality with automatic environment reconstruction, doi:10.1007/s10055-010-0172-8. *Virtual Reality Journal*, 1–14.

Asada, H., & Izumi, H. (1989). Automatic program generation from teaching data for the hybrid control of robots. *IEEE Transactions on robotics and automation*, *5*(2), 166–173.

Brown, J., & Burton, R. (1978). Diagnostic models for procedural bugs in basic mathematical skills. *in Cognitive Science*, *2*(2), 155–192.

Buehler, M., Koditschek, D., & Kindlmann, P. (1994). Planning and control of robotic juggling and catching tasks. *International Journal of Robotics Research*, *13*(2), 101–108.

Chamaret, D., Ullah, S., Richard, P., & Naud, M. (2010). Integration and evaluation of haptic feedbacks: from cad models to virtual prototyping. *International Journal on Interactive Design and Manufacturing IJIDEM10*, *4*(2), 87–94.

Chen, Y., & Naghdy, F. (2002). Human to robot skill transfer through haptic rendered environment. In *Proceedings of australasian conference on robotics and automation acra* (pp. 197–202).

Chong, J., Ong, S., Nee, A., & Youcef-Youmi, K. (June 2009). Robot programming using augmented reality: An interactive method for planning collision-free paths. *Robotics and Computer-Integrated Manufacturing*, *25*(3), 689–701.

Cooper, S., Hertzmann, A., & Popovic, Z. (2007). Active learning for real-time motion controllers. In *Proceedings of acm siggraph'07, acm transactions on graphics (tog)* (Vol. 26).

Delson, N., & West, H. (1994). Robot programming by human demonstration: the use of human variation in identifying obstacle free trajectories. In *Proceedings of the ieee international conference on robotics and automation (icra)* (Vol. 1, pp. 564–571).

Dillmann, R., Rogalla, O., Ehrenmann, M., Zollner, R., & Bordegoni, M. (1999). Learning robot behaviour and skills based on human demonstration and advice: the machine learning paradigm. In *9th international symposium of robotics research (isrr)* (Vol. 9, pp. 229–238).

Dreyfus, G., Martinez, J., Samuelides, M., & Gordon, M. (2002). *Neural network: methods and applications*. Eyrolles.

Friedrich, H., Grossmann, V., Ehrenmann, M., Rogalla, O., Zollner, R., & Dillmann, R. (1999). Towards cognitive elementary operators: grasp classification using neural network classifiers. In *Proceedings of international conference on intelligent systems and control, iasted* (pp. 121–126).

Hirai, H., & Miyazaki, F. (2005). Dynamic temporal patterns between robots: The self-organized timing selection in juggling-like passing the balls. In *Proceedings ieee international conference on robotics and biomimetics, (robio 2005)* (pp. 511–516).

Hove, B., & Slotine, J. (1991). Experiments in robotic catching. In *Proceedings of the american control conference* (Vol. 1, pp. 380–385).

Kawasaki, H., Furukawa, T., Ueki, S., & Mouri, T. (2008). Virtual robot teaching in the assembly work environment for multi fingered robots. In *Proceedings of automation congress, wac 2008* (pp. 1–7).

Kawasaki, H., Nakayama, K., Mouri, T., & Ito, S. (2001). Virtual teaching based on hand manipulability for multi-fingered robots. In *Proceedings ieee international conference on robotics and automation* (Vol. 2, pp. 1388–1393).

Knoop, S., Pardowitz, M., & Dillmann, R. (2008). From abstract task knowledge to executable robot programs. *in Journal of Intelligent and Robotic Systems: Theory and Application*, *52*.

Kuniyoshi, Y., Inaba, M., & Inoue, H. (1994). Learning by watching: Extracting reusable task knowledge from visual observation of human performance. In *Ieee transaction on robotics and automation* (Vol. 10, pp. 799–822).

Miyazaki, F., & Mori, R. (2004). Realization of ball catching task using a mobile robot. In *Proceedings of ieee international conference on networking, sensing and control* (Vol. 1, pp. 58–63).

Paul, R. (1979). *The theory and practice of robot manipulator: programming and control*.

Payeur, P., Le-Huy, H., & Gosselin, C. (1995). Trajectory prediction for moving objects using artificial neural networks. *IEEE Transactions on Industrial Electronics*, *42*(2), 147–158.

Richard, P., Burdea, G., Birebent, G., Gomez, D., Langrana, N., & Coiffet, P. (1996). Effect of frame rate and force feedback on virtual objects manipulation. *Presence - Teleoperators and Virtual Environments (MIT Press)*, *15*, 95–108.

Richard, P., Chamaret, D., Inglese, F., Lucidarme, P., & Ferrier, J. (2006). Human-scale haptic virtual environment for product design: Effect of sensory substitution. *International Journal of Virtual Reality*, *5*(2), 37–44.

Riley, M., & Atkeson, C. (2002). Robot catching : Towards engaging human-humanoid interaction. *in Autonomous Robots*, *12*(1), 119–128.

Rizzi, A., & Koditschek, D. (1992). Progress in spatial robot juggling. In *Proceedings ieee international conference on robotics and automation* (Vol. 1, pp. 775–780).

Rizzi, A., & Koditschek, D. (1993). Further progress in robot juggling. the spatial two-juggle. In *Proceedings of the ieee international conference on robotics and automation* (Vol. 3, pp. 914–924).

Rose III, C., Sloan, P., & Cohen, M. (2001). Artist-directed inverse-kinematics using radial basis function interpolation. In *Computer graphics forum* (Vol. 20, pp. 239–250).

Schraft, R., & Meyer, C. (2006). The need for an intuitive teaching method for small and medium enterprises. In *International symposium on robotics* (Vol. 1956).

Spong, M., & Vidyasagar, M. (1989). *Robot dynamics and control*. John Wiley and Sons.

Tsubone, T., Kurimoto, K., Sugiyama, K., & Wada, Y. (2008). Task learning based on reinforcement learning in virtual environment. In *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)* (Vol. 4985, pp. 243–253).

Webb, G., Pazzani, M., & Billsus, D. (2001). Machine learning for user modelling. In *User modeling and user-adapted interaction* (Vol. 11, pp. 19–29).

Wojtara, T., & Nonami, K. (2004). Hand posture detection by neural network and grasp mapping for a master slave hand system. In *Ieee/rsj international conference on intelligent robots and systems (iros)* (Vol. 1, pp. 866–871).

Yutuo, C., Xuli, H., Okada, M., Chen, Y., & Naghdy, F. (2007). Intelligent robotic peg-in-hole insertion learning based on haptic virtual environment. In *Proceedings of 10th ieee international conference on computer aided design and computer graphics, cad/graphics* (pp. 355–360).