In Proceedings of Intelligent Virtual Agents (IVA), Boston, 2014
(This version is the authors' manuscript, the final publication is available at link.springer.com)

Motion Parameterization and Adaptation Strategies for Virtual Therapists

Carlo Camporesi[1], Anthony Popelar[1], Marcelo Kallmann[1], and Jay Han[2]

[1]School of Engineering, University of California Merced
[2]Department of Physical Medicine and Rehabilitation, University of California Davis

Abstract. We propose in this paper new techniques for correction and parameterization of motion capture sequences containing upper-body exercises for physical therapy. By relying on motion capture sequences we allow therapists to easily record new patient-customized exercises intuitively by direct demonstration. The proposed correction and parameterization techniques allow the modification of recorded sequences in order to 1) correct and modify properties such as alignments and constraints, 2) customize prescribed exercises by modifying parameterized properties such as speed, wait times and exercise amplitudes, and 3) to achieve real-time adaptation by monitoring user performances and updating the parameters of each exercise for improving the therapy delivery. The proposed techniques allow autonomous virtual therapists to improve the whole therapy process, from exercise definition to delivery.

Keywords: virtual therapists, motion capture, virtual humans.

1 Introduction

The motivation of this work is to improve the usability of virtual humans serving as virtual therapists autonomously delivering physical therapy exercises to patients. We focus on the problem of automatic correction and parameterization of motion capture sequences defining upper-body exercises. Customized exercises per patient can be intuitively recorded from therapists by direct demonstration using the Kinect sensor or any other suitable motion capture device. Given a captured exercise, we propose correction and parameterization techniques that allow 1) fine-tuning of key characteristics of the exercise such as alignments and constraints, 2) customization of the exercises by modifying parameterized properties such as speed, wait times and amplitudes, and 3) real-time adaptation by monitoring user performances and updating exercise parameters in order to improve therapy delivery.

The presented techniques greatly facilitate the process of defining exercises by demonstration, allowing the customization of exercises to specific patients. We focus on providing parameterization while at the same time reproducing, to the desired degree, any small imperfections that are captured in the motion in order to maintain the humanlike behavior of the virtual therapist during therapy delivery. As a result the proposed methods produce realistic continuous motions

Fig. 1. Illustration of the system being used in practice.

that can adapt to user responses in order to improve the overall experience of performing the exercises.

2 Related Work

The use of new technologies to overcome the limitations of standard approaches to physiotherapy is becoming increasingly popular. A typical approach in some applications is to track user movements while a virtual character displays the exercises to be executed. The representations of the user and virtual trainer are usually displayed side by side or superimposed to display motion differences, improving the learning process and the understanding of the movements [5, 16].

Automated systems often allow parameterization capabilities. For instance, Lange et al. [7] describe core elements that a VR-based intervention should address, indicating that clinicians and therapists have critical roles to play and VR systems are tools that must reflect their decisions in terms of a person's ability to interact with a system, types of tasks, rates of progression, etc [8, 4].

The key benefit of adopting a programming by demonstration approach is to allow the intuitive definition of new exercises as needed. The overall approach has been adopted in many areas [2, 14, 10], and it involves the need to automatically process captured motions according to the goals of the system.

Velloso et al. [15] propose a system that extracts a movement model from a demonstrated motion to then provide high-level feedback during delivery, however without motion adaptation to the user performances. The YouMove system [1] trains the user through a series of stages while providing guidance, however also not incorporating motion adaptation to the user performances.

We propose new motion processing approaches to achieve adaptive motions that are both controllable and realistic. While motion blending techniques with motion capture data [12, 6, 11, 2] provide powerful interpolation-based approaches for parameterizing motions, they require the definition of several motion examples in order to achieve parameterization. In contrast our proposed techniques are simple and are designed to provide parameterization of a given single exercise motion. We rely both on structural knowledge of exercises and on generic constraint detection techniques, such as detection of fixed points [9, 13] and motion processing with Principal Component Analysis (PCA) [3].

3 Detection of Constraints and Parameterizations

Given a new exercise motion demonstrated to the system, the system will analyze the motion and detect the parameterizations that can be employed. An input motion is represented as a collection of frames $M_i, i \in \{1, \ldots, n\}$, where each frame M_i is a vector containing the position and the joint angles that define one posture of the character in time.

3.1 Detection of Geometrical Constraints

Our constraint detection mechanism is designed for three specific purposes: to inform motion parameterization, to help correcting artifacts and noise in the motions, and to provide metrics for quantifying motion compliance. The metrics are used to provide visual feedback to the user, to inform the correctness of performed motions, to make decisions during the real-time adaptation mechanism, and to achieve an overall user performance score for each session.

Appropriate constraints are not constraints which are to be absolutely followed. Recorded motions may have unintended movements and imperfections introduced by the capture system. Constraints must be detected despite these fluctuations, and should be softly enforced so the motion can be made to look correct and also natural.

We analyze the position in space of a specific joint with respect to a frame of reference F which can be placed at any ancestor joint in the skeleton structure. The detection framework can accommodate any desired type of constraint but in this paper we focus on two types of constraints: Point and Planar.

- A Point Constraint (Figure 2) describes a child joint that is static relative to its parent. Let's $P_i, i \in \{l, \ldots, k\}$ be the cloud of points formed by a joint trajectory with respect to a local frame F generated by resampling linearly the motion frames with constant frame rate. The standard deviation of the cloud of points σ is calculated and subsequently checked against a specific threshold α. When the condition is met the current joint is marked as a point constraint and it is represented by the specific point located at the mean μ. When a point constraint is detected the ancestor(s) can be then adjusted to enforce the constraint.

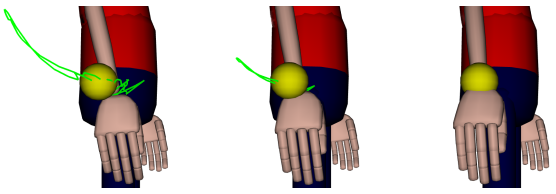

Fig. 2. Point Constraint. The yellow sphere represents the detection of a point constraint at the elbow joint. From left to right: the wrist motion trajectory (depicted in green) is corrected to the mean point with 0%, 50%, and 100% correction.

• A Plane Constraint (Figure 3) detects if a joint moves approximately within a plane. Similarly to the point constraint detection a point cloud is first generated. Then, PCA is applied to the set of points to determine a proper orthogonal decomposition considering the resulting Eigenspace from the covariance. A planar surface is then retrieved considering the two Eigenvectors with higher Eigenvalue λ (the magnitude of λ is used to validate the plane). The average distance of the points from this plane is then checked against a threshold β to determine if a plane constraint is appropriate for the given joint.

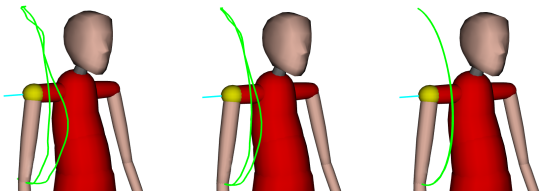

Fig. 3. Plane Constraint. The blue axis is the normal direction of the detected plane constraint affecting the shoulder joint. From left to right: correction level (from 0% to 100%) where the elbow trajectory (green trajectories) is gradually collapsed into a plane.

3.2 Geometrical Constraint Alignment

Let i be the index of the frame currently evaluated. Let p_i be the position in space of the current joint and q_i be a quaternion representing the current local orientation. A point constraint is defined considering the orientation q_m in the local orientation frame that represents the vector defined by the local point constraint. A point constraint is enforced through spherical linear interpolation between q_i and q_m. Figure 2 shows the trajectories generated by the wrist joint collapsing into a point constraint.

To apply the plane constraint, we identify the orientation defined by the projection of each point p_i to the plane discovered during the detection phase. The plane constraint is then enforced, similarly to the point constraint, by interpolating the equivalent orientations. Figure 3 shows the trajectories generated by the elbow joint aligning into a plane constraint.

3.3 Detection of Exercise Parameterization

Consider a typical shoulder flexion exercise where the arm is raised until it reaches the vertical position or more (initial phase); subsequently the arm is hold for a few seconds (hold phase) and then it relaxes back to a rest position (return phase). This is the type of exercise that we seek to parameterize.

The analysis procedure makes the following assumptions: a) each motion M represents one cycle of a cyclic arm exercise; b) the first frame of a motion contains a posture representing the starting point of the exercise; c) the exercise will have distinct phases: the initial phase (M_{init}) is when the arm moves from the initial posture towards a posture of maximum exercise amplitude, then the exercise may or not have a hold phase (M_{hold}) but at some point the exercise

must enter the return phase (M_{end}), where the exercise returns to a posture similar to the starting posture. In addition, if the motion contains a hold phase at the point of maximum amplitude, it will mean that an approximately static pose of some duration (the hold phase duration) exists at the maximum amplitude point. We also consider an optional 4$^{\text{th}}$ phase that can be added to any exercise, the wait phase (M_{wait}), which is an optional period of time where the character just waits in its rest pose before performing a new repetition of the exercise. Figure 4 illustrates a typical exercise that fits our assumptions.

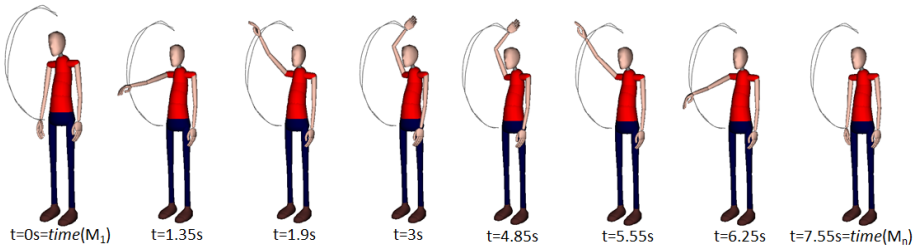

Fig. 4. Example of a typical exercise captured from a therapist in one of our tests with the system. The shown trajectory is the trajectory of the right wrist joint along the entire motion. The initial phase happens between t=0s and t=3s. Then, between t=3s and t=4.85s there is a hold phase at maximum amplitude where the therapist is static (but small posture variations are always noticeable). Then, between t=4.85s and t=7.55s we can observe the return phase.

The analysis if the exercise can be parameterized has two main steps: first the arm to be parameterized is detected; and then the two *motion apices* are detected. The apices, or the points of maximum amplitude, are the intersection points between the initial and return phases with the hold phase (frames $t = 3s$ and $t = 4.85$ in Figure 4). These points will be a single apex point if the motion has no hold phase in it. If the phases above are executed successfully the input motion is segmented in initial, return and an optional hold phase, and the motion can be parameterized.

In order to detect which arm to parameterize we extract the global positions of the left and right wrists along their trajectories. Let L_i and R_i respectively denote these positions. Since our focus is on arm exercises the wrist represents an obvious distal joint of the arm kinematic chain to use in our parameterization analysis algorithm. For each wrist trajectory L and R we compute the 3D bounding box of the 3D trajectory. The bounding box dimension is used to determine which arm is moving and if the motion can be parameterized. As a result of this process, the analysis will return one of the following four options: a) the motion cannot be parameterized; b) the motion will be parameterized by the left/right arm; or d) the motion will be parameterized by both arms (targeting symmetrical exercises).

4 Exercise Parameterization

If the motion can be parameterized and its type is determined, we then search the motion for the points of maximum amplitude. To detect one apex point we search for a frame that indicates a sharp turn in trajectory. Since the motion may or not contain a hold phase, we perform the search in two steps: a forward search starting from M_1, and a backward search starting from M_n.

Let i be the index of the current frame being evaluated (M_i). Let T represent the trajectory of the left or right wrist joint, that is, T_i will be R_i or L_i (the trajectory is first smoothed through moving mean to reduce sensor noise). In order to determine if M_i represents an apex point we perform the following steps. We discard the initial points until the distance between two consecutive points becomes greater than a specific threshold d_t (a threshold of 5cm worked well in practice). We first compute the incoming and outgoing direction vectors with respect to T_i, respectively: $a = T_i - T_{i-1}$, and $b = T_{i+1} - T_i$. If a or b is a null vector, that means we are in a stationary pose and we therefore skip frame M_i and no apex is detected at position i. Otherwise, the angle α between vectors a and b is computed and used to determine if there is a sharp change in direction at position i. If α is greater than a threshold angle, frame i is considered a probable apex point, otherwise we skip and proceed with the search. We are using a threshold of 75 degrees and this value has worked well in all our examples with clear detections achieved. To mark an apex to be definitive we consider the distance between the following k frames to be less than d_t.

The test described above is first employed for finding the first apex point by searching forward all frames (starting from the first frame). The first apex found is called Apex 1 and its frame index is denoted as a_1. If no apex is found the motion cannot be parameterized. If Apex 1 is successfully found, then the search is employed backwards starting from the last frame, however not allowing passing beyond Apex 1. The second apex found is called Apex 2 (a_2). Note that Apex 2 may be the same as Apex 1, in which case no holding phase is present in the input motion. After the described analysis, the main three portions of the motion have been detected: a) the initial phase is defined by frames $\{1, 2, \ldots, a_1\}$ (motion segment M_{init}); b) the hold phase is defined by frames $\{a_1, a_1 + 1, \ldots, a_2\}$, if $a_2 > a_1$, and nonexistent otherwise; and c) the return phase is defined by frames $\{a_2, a_2 + 1, \ldots, n\}$ (motion segment M_{ret}). Once an input motion M is successfully segmented, it can then be parameterized.

Amplitude and Hold Phase Parameterization We parameterize amplitude in terms of a percentage of the wrist trajectory: 100% means that the full amplitude observed in the input motion M is to be preserved, if 80% is given then the produced parameterized motion should go into hold or return phase when 80% of the original amplitude is reached, and so on. Let h be the time duration in seconds of the desired hold duration. When the target amplitude is reached, the posture at the target amplitude is maintained for the given duration h of the desired hold phase. When the hold phase ends, the posture is blended into the return motion M_{ret} at the current amplitude point towards the final frame of M_{ret}. See Figure 5. The described operations are enough to

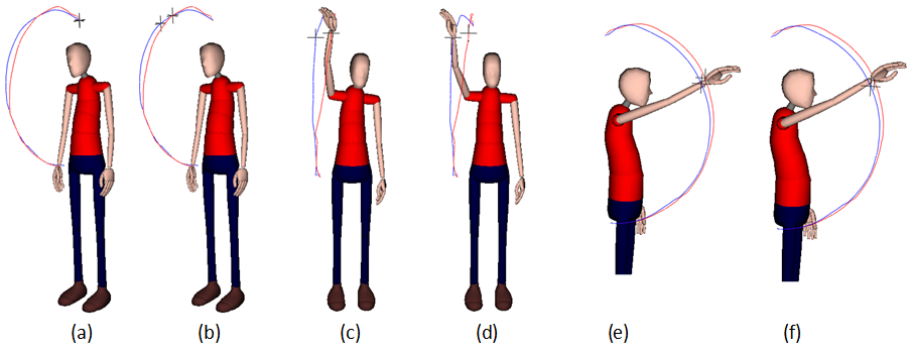

Fig. 5. The red trajectory shows the initial phase M_{init}. The blue trajectory shows the return phase M_{ret}. The input motion is the same as Figure 4. (a) The full (100%) amplitude of the input motion is shown by the trajectories. Two black crosses at the end of the trajectories (in almost identical positions) mark the positions of Apex 1 and Apex 2. (b) The two black crosses now mark the maximum amplitude points in the initial and return trajectories at 75% amplitude. (c,d) In this frontal view it is possible to notice that the postures at 75% amplitude in the initial and return phases are different. The hold phase will start by holding the posture shown in (c), and when the hold phase is over, we blend into the return motion at the posture shown in (d) in order to produce a smooth transition into the return phase. (e,f) Lateral view.

achieve a continuous parameterized motion, however two undesired effects may happen: a noticeable abrupt stop of M_{init} or an unnatural start of M_{ret}, because the parameterization may suddenly blend motions to transition at points with some significant velocity. To correct this we re-time the segments so that motion phases always exhibit ease-in or ease-out profiles.

Behavior During Hold and Wait Phases In order to improve the realism, we add a small oscillatory spine movement mimicking a breathing motion, which is applied to spine joints during the hold and wait phases. One particular problem that is addressed here is to produce an oscillatory motion that ends with no contribution to the original pose at the end of the oscillation period. This is needed so that, after the oscillation period, the motion can naturally continue towards its next phase and without additional blending operations. We thus have to produce oscillations of controlled amplitude and period. This is accomplished with the following function: $f(t) = d\sin(t\pi/d)$, if $d < 1$, and $\sin(t\pi/(d/floor(d)))$ otherwise; where $d > 0$ is the duration of the oscillation period, which in our case will be the duration of the hold or wait periods.

At the beginning of a hold or wait phase we save the joint angles of the spine in a vector s, and then apply to the spine joints the values of $s + cf(t)$, where $t \in [0, d]$, and c is an amplitude constant. We obtained good behavior with $c = 0.007$, and only operating on one degree of freedom of two spine joints: one near the root of the character hierarchy, and one about the center of the torso. The used degree of freedom is the one that produces rotations on the sagittal

plane of the character. The achieved breathing behavior can be observed in the video accompanying this paper.

Overall Parameterization The described procedures allow us to parameterize an input motion M with respect to up to four parameters: amplitude a (in percentage), hold time h (in seconds), wait time w (in seconds), and speed s (as a multiplier to the original time parameterization). Given a set of parameters (a, h, w, s), the input motion can be prepared for parameterization very efficiently and then, during execution of the parameterized motion, only trivial blending operations are performed in real-time.

5 Real-Time Adaptation

When the adaptation mechanism is enabled the system collects information from the patient's performance in real-time and adapts the current exercise in its next repetition. In addition, visual feedback is also provided: arrows showing direction of correction for improving motion compliance, constraint violation feedback and also an overall performance score with explanatory text (see Figure 6).

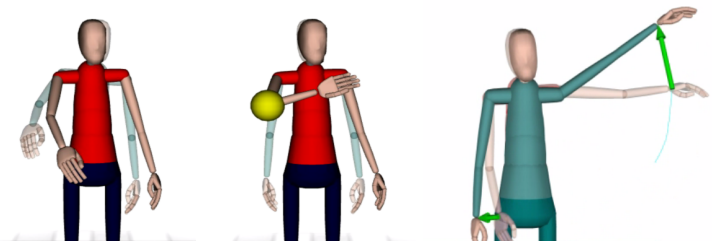

Fig. 6. The red character displays the user's motion and the blue one the target exercise. Left: no violated constraints. Center: user is reminded to correct the elbow. Right: arrows show direction of correction to improve compliance.

Four types of adaptation mechanisms are provided:

- **Amplitude Adaptation** The range can vary from 75% to 100% of the target amplitude. The system tracks the distance between the patient's active end-effector and the apex at the target amplitude position. If the minimum distance is larger than the amplitude compliance parameter specified by the therapist, the next exercise execution will have the target amplitude lowered to become within the compliance range. If in a subsequent repetition the user reaches the current (reduced) target amplitude, then the next target amplitude will be increased towards the original target amplitude.
- **Hold Time** The hold phase adaptation is designed to adapt the time at hold stance to improve resistance, usually in a posture that becomes difficult to maintain over time. The maximum distance between the target hold point and

the performed end-effector position is computed. If above a threshold, the patient is having difficulty in maintaining the posture and the next exercise repetition will have a shorter hold phase duration. If in a subsequent repetition the patient is able to well maintain the hold posture, then the hold duration is gradually increased back to its previous value.

• **Speed Execution** During patient monitoring, the active position of the patient's end-effector is tracked and its distance to the demonstrated exercise end-effector is computed for every frame. If the average distance computed across the entire exercise is above a given trajecory compliance threshold (see Figure 7), the next exercise execution speed is decreased. If in a subsequent repetition the difference is under the threshold the speed will be gradually adjusted back.

• **Wait-Time Between Exercises** The initial wait time specified by the therapist is decreased or increased in order to allow the patient to have an appropriate time to rest between exercises. A performance metric based on averaging the trajectory compliance and the hold phase completion metrics is used to determine how well the patient is being able to follow an exercise. If the user is well performing the exercises a shorter wait time is selected, otherwise a longer wait time is preferred. In this way wait times are related to the experienced difficulty in each exercise, and they adapt to specific individuals and progress rates.

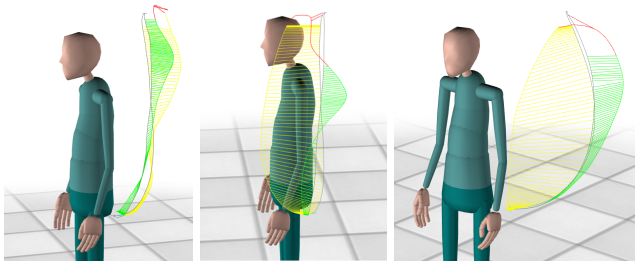

Fig. 7. From left to right: high, medium and low trajectory compliance.

6 Results and Conclusions

The described algorithms have been tested for constraint detection and motion parameterization with a variety of arm exercises and different users. The obtained results were always as expected, within reasonable compliance with the defined exercise structure. All presented methods are very efficient for real-time computation. While this paper focuses on motion processing techniques, the described adaptation strategies have been specified from many discussions with therapists according to their needs while experimenting with our prototype system. Many variations and adjustments are possible, a final version will only be determined after the system and its several features are evaluated in practice.

A supplemental video is available to demonstrate the obtained results. In summary, we present contributions on new motion processing approaches for single motion parameterization, and as well on novel strategies for motion adaptation to users during real-time exercise delivery.

Acknowledgments This work was partially supported by NSF Award CNS-1305196 and by a HSRI San Joaquin Valley eHealth Network seed grant funded by AT&T.

References

1. Anderson, F., Grossman, T., Matejka, J., Fitzmaurice, G.W.: YouMove: enhancing movement training with an augmented reality mirror. In: Proceedings of User Interface Software and Technology (UIST). pp. 311–320. ACM (2013)
2. Camporesi, C., Huang, Y., Kallmann, M.: Interactive motion modeling and parameterization by direct demonstration. In: Proceedings of the 10th International Conference on Intelligent Virtual Agents (IVA) (2010)
3. Glardon, P., Boulic, R., Thalmann, D.: A coherent locomotion engine extrapolating beyond experimental data. In: Proceedings of Computer Animation and Social Agent. pp. 73–84 (2004)
4. Grealy, M., Nasser, B.: The use of virtual reality in assisting rehabilitation. Advances in Clinical Neuroscience and Rehabilitation 13(9), 19–20 (2013)
5. Holden, M.K.: Virtual environments for motor rehabilitation: review. Cyberpsichology and Behavior 8(3), 187–211 (2005)
6. Kovar, L., Gleicher, M.: Automated extraction and parameterization of motions in large data sets. ACM Transaction on Graphics 23(3), 559–568 (2004)
7. Lange, B., Koenig, S., Chang, C.Y., McConnell, E., Suma, E., Bolas, M., Rizzo, A.: Designing informed game-based rehabilitation tasks leveraging advances in virtual reality. Disability and rehabilitation 34(22), 1863–1870 (2012)
8. Levac, D.E., Galvin, J.: When is virtual reality therapy? Archives of Physical Medicine and Rehabilitation 94(4), 795–798 (2013)
9. Liu, C.K., Popović, Z.: Synthesis of complex dynamic character motion from simple animations. ACM Trans. Graph. 21(3), 408–416 (Jul 2002)
10. Lü, H., Li, Y.: Gesture coder: a tool for programming multi-touch gestures by demonstration. In: Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems. pp. 2875–2884. ACM (2012)
11. Ma, W., Xia, S., Hodgins, J.K., Yang, X., Li, C., Wang, Z.: Modeling style and variation in human motion. In: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA) (2010)
12. RoseIII, C.F., Sloan, P.P.J., Cohen, M.F.: Artist-directed inverse-kinematics using radial basis function interpolation. Computer Graphics Forum (Proceedings of Eurographics) 20(3), 239–250 (September 2001)
13. Salvati, M., Le Callennec, B., Boulic, R.: A Generic Method for Geometric Contraints Detection. In: Eurographics (2004)
14. Skoglund, A., Iliev, B., Palm, R.: Programming-by-demonstration of reaching motions - a next-state-planner approach. Robotics and Aut. Systems 58(5) (2010)
15. Velloso, E., Bulling, A., Gellersen, H.: Motionma: Motion modelling and analysis by demonstration. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. pp. 1309–1318. CHI '13, ACM, New York, NY, USA (2013)
16. Wollersheim, D., Merkes, M., Shields, N., Liamputtong, P., Wallis, L., Reynolds, F., Koh, L.: Physical and psychosocial effects of Wii video game use among older women. Intl Journal of Emerging Technologies and Society 8(2), 85–98 (2010)