STUDIA INFORMATICA 2003
Volume 24 Number 4 (56)

Jarosław FRANCIK, KATARZYNA TRYBICKA-FRANCIK
Politechnika Śląska, Instytut Informatyki

A FRAMEWORK FOR PROGRAM CONTROL OF ANIMATION OF HUMAN AND ANIMAL CHARACTERS[*]

Summary. Our objective is to add a significant level of automation to the process of animating human and animal avatars using commercially available animation packages. One of possible solutions is controlling the animation in an external unit. The proposed framework, called KINE+ supports importing a 3D model data from an animation package, a bone-based animation control and, finally, exporting the generated motion definitions back to a 3D package. Two applications have been also presented.

Słowa kluczowe: animation, 3D model, avatar, collision detection

ARCHITEKTURA DLA POTRZEB AUTOMATYCZNEGO STEROWANIA ANIMACJĄ POSTACI LUDZKICH I ZWIERZĘCYCH

Streszczenie. Naszym celem jest wprowadzenie znaczącego poziomu automatyzacji do procesu animacji ludzi i zwierząt za pomocą dostępnych na rynku pakietów 3D. Jednym z możliwych rozwiązań jest sterowanie animacją w zewnętrznym module. Zaproponowana architektura, KINE+, wspomaga pozyskiwanie danych o modelu 3D z pakietu 3D, sterowanie animacją oparte na systemie kości oraz eksport utworzonych opisów ruchu z powrotem do pakietu 3D. Przedstawiono też dwa zastosowania stworzonych narzędzi.

Słowa kluczowe: animacja, model 3D, awatar, wykrywanie kolizji

1. Introduction

The animation of three-dimensional characters (avatars) became a relevant issue in the industry for film, video, games and special effects for post production. The commercial animation packages such as 3D StudioMax™ or Maya™ are widely used by the professionals [1-3].

[*] Research supported by Silesian Univ. of Technology Project no. BW-486/RAu-2/2002

They contain three principal components (fig. 1):

- the modeller,
- the animation creator,
- the renderer.

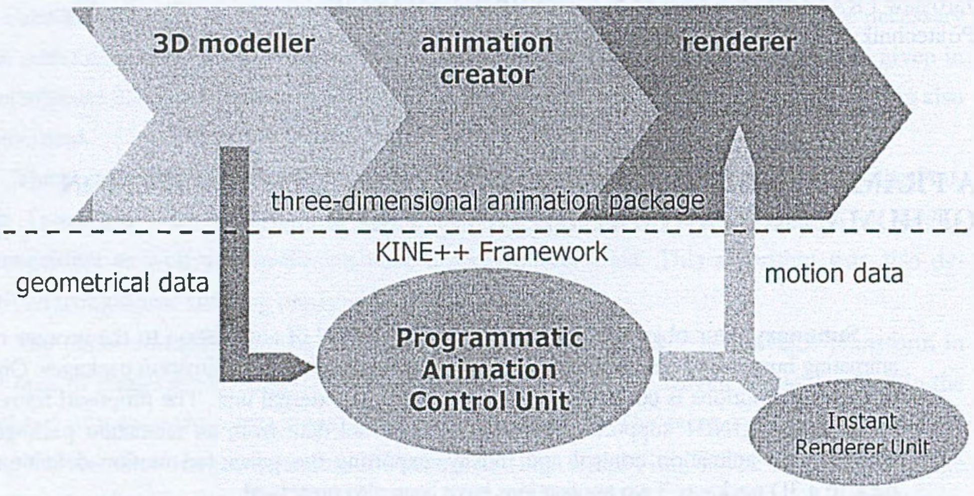

Fig. 1. Principal components of 3D packages and the proposed external processing architecture
Rys. 1. Główne komponenty pakietu 3D i proponowana architektura zewnętrznego przetwarzania

The modeller enables the user to define 3D models of objects or characters. It is done by determining their geometric structure and surface characteristics. The animation creator is then used to define the animation key frames by placing the modelled objects and characters within a 3D scene. Finally, after defining the lighting and virtual camera positions, the renderer generates the animated sequence between given key frames.

So far, it is only the renderer that provides true automation of the process. Both modeller and animation creator facilitate, but do not automate the user's tasks. Although recent versions of the animation packages have incorporated limited environment awareness and some collision avoidance, there is still plenty of scope for improvements, including environment-aware motion planning, high-level interaction and other AI-based issues.

The animation of three-dimensional characters is even more challenging than animation of any other objects [4, 5]. The aim of the presented works is to provide a significant level of additional automation to the process of animating human and animal avatars. This will be obtained by providing an additional, programmatic step of processing done outside the 3D package, in an external animation control unit (fig. 1). This paper presents an architecture that makes it possible to acquire geometrical data on a 3D model created with a 3D package, produce and control animation on that model and, finally, export the animation – in form of a motion definition file – back to the 3D package. There is no support for 3D modelling or final rendering: these two phases should be done in the 3D package, however a renderer for instant visualization may be added, at least for a quick preview of animation results (actually a limited version of such unit is already implemented).

2. Background

Animation of avatars attracted much attention [4-5]. Any effort in the area must have a deep background in biomechanics [6]. Other important issues embrace forward and inverse kinematics [7] and collision detection and avoidance techniques [8]. Nevertheless many problems remain still unsolved. An interaction between two or more characters or between a character and any external objects seems to be of particular interest.

If any kind of a procedural animation control is taken into account (like [9]), an efficient model of the avatar's (bio)mechanics is essential. Such models are usually based on use of skeletons (fig. 2, first on the left). Skeleton is the simplest abstraction of a 3D character: it's a hierarchical system of smaller parts called bones (tab. 1). Each bone can be treated as a segment in 3D space, defined with a pair of points (often called joints). Position and orientation of a bone is defined relatively to its "superbone" that is above it in the hierarchy, by a set of Euler angles, transformation matrix or a quaternion. So, to compute the absolute position and location of a given bone, it is necessary to apply transformations recursively across the whole hierarchy to the root (which should be defined in absolute values). Transformation of any bone affects all its derived bones, ie. a rotation of an upper arm affects the whole arm. For a given bone, its translation (relative position) is an inherent feature of the model, while its rotation (relative orientation) may be freely changed to model bending the joints.

The next step after the skeleton is a hierarchy of oriented bounding boxes (OBB, fig. 2, second on the left). Each bone is bound in a OBB, which is the smallest box that entirely contains a body part connected with this bone. OBB's are usually used to detect collisions [10] (between avatar's different parts or between avatar and other objects). Mapping each bone to a single BB is enough for many applications, however when more precision is necessary, OBB's may be divided into a hierarchy of smaller boxes to better conform the shape [10].

Its final shape an avatar obtains by the process of skinning: fitting the external surface (skin) to the internal skeleton. Sometimes an intermediary stage is also used. In the Character Studio™ package [11] (a plug-in product for the 3D StudioMax™ [12]), such intermediary 3D object built on top of a system of bones is called *biped* (fig. 2, second on the right). Its shape is closer to the final avatar's shape, it's more comfortable to animate than the skeleton, and it's a good starting point for the final skinning. In either case, fitting a final skin to an existing skeleton or biped is relatively easy (fig. 2, first on the right). Once the animation is designed on the skeleton/biped level, it can be applied for the whole skinned avatar automatically.

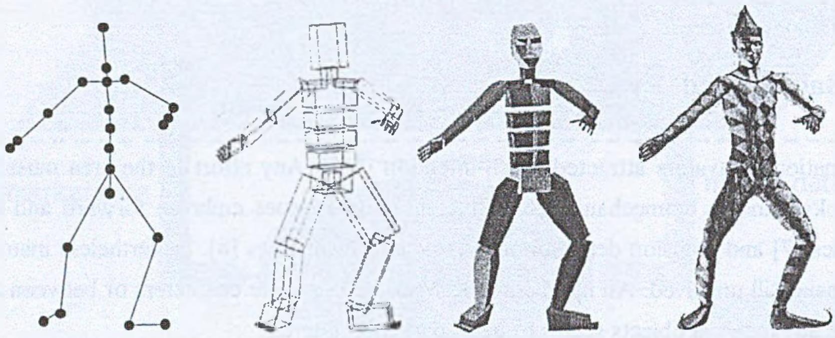

Fig. 2. Skeleton (w/o fingers or toes), bounding boxes, biped and skinned avatar for a sample character
Rys. 2. Szkielet (bez palców), zestaw *bounding boxes*, *biped* (manekin) oraz awatar ze skórą dla przykładowej postaci ludzkiej

Table 1
Bone hierarchy of an avatar of fig. 2

```
pelvis
├─ spine0
├─ left thigh
│  └─left calf
│     └─ left foot
│        ├─ left toe0 (3 links)
│        │  ...
│        └─left toe4 (3 links)
├─right thigh
│  └─ right calf
│     └─right foot
│        ├─right toe0 (3 links)
│        │  ...
│        └─left toe4 (3 links)
└─ spine1
   └─ spine2
      └─ spine3
         └─ neck
            ├─ head
            ├─ left clavicle
            │  └─ left upper arm
            │     └─ left forearm
            │        └─ left hand
            │           ├─ left finger0 (3 links)
            │           │  ...
            │           └─ left finger4 (3 links)
            └─ right clavicle
               └─ right upper arm
                  └─ right forearm
                     └─ right hand
                        ├─ right finger0 (3 links)
                        │  ...
                        └─ right finger4 (3 links)
```

3. The KINE+ framework

KINE+, a framework presented here, implements a full skeleton of a human or animal avatar. The aim was to make programmatic control of the avatar animation available to C++ programmers (other languages are also supported). A capability of importing and exporting 3D models from and to a professional 3D package is essential. No skinning is implemented: as we stated in previous section, the animation is usually designed on the level of a skeleton or a biped. However a relatively easy real-time instant rendered is available just for a quick preview.

3.1. Importing the Geometry

Currently the framework supports data imported from the Discreet Character Studio™ [11]. This product is a plug-in for worldwide standard 3D package, 3D StudioMax™ [12]. It extends 3D StudioMax™ functionality with bipeds – fully featured, skeleton based human and animal avatars (fig. 2).

The KINE+ bone hierarchy is the same as the one used by Character Studio™ bipeds. Main bones are shown in tab. 1; there are several more optionally supported bones, for example additional segments of the head and the neck, the tail (up to 5 links) or two ponytails (a "tail" made of hair). All this makes it possible to create different biped figures just by adjusting the number and the length of bones (fig. 3). All the adjustable parameters are simply transferred to KINE+ in form of a native-format binary file, together with OBB data.

3.2. Controlling the Animation

The geometry file may be loaded at runtime or statically linked to the code. Once the geometry is loaded, the KINE+ framework is ready to work. From the point of view of the user it is just a library available in a form of C++ LIB/H files or as an in-process COM+ component. This makes KINE+ available for almost every programming environment in Windows.

KINE+ supports following animation related features:

- Geometrical set-up: by importing the geometry – as described in the previous subsection – or defining it directly by setting various skeleton parameters.
- Forward kinematics (FK). Motion may be specified by defining relative rotations of the chosen bones (along with the time stamp). Rotations may be given in a form of Euler angles, rotation matrices or quaternions. FK data are used to set-up key frames; interpolating of the frames in-between is done automatically when necessary.
- Inverse kinematics (IK). It is an alternative method of key frame set-up: there are rather spatial co-ordinates of joints specified instead of relative rotations. All the rota-

tion matrices are then retrieved automatically by KINE+. This approach is more comfortable in many applications. Currently the IK algorithm applied is relatively poor, it sometimes leads to unfortunate motions. Soon there will be available a much more sophisticated approach [7].

- Position reading. In any moment of animation it is possible to get the spatial coordinates and/or transformation matrix (Euler angles, quaternion) of any bone.

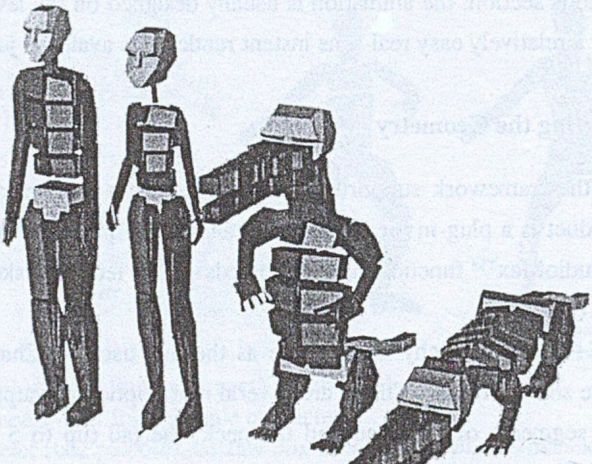

Fig. 3. Various characters created on the same skeleton schema (standard figures distributed with Character Studio™)
Rys. 3. Różne postacie utworzone na podstawie tego samego schematu szkieletu (standardowe sylwetki pakietu Character Studio™)

- Collision detection and avoiding. Currently collision detection is available for key frames only. It is sufficient for most applications; if greater precision is needed, additional key frames may be additionally added. Collision detection for every frame is not supported; in any case it would be very time-consuming. Currently the framework allows deciding if there is a collision or not; one of the future work is to make it possible to find out the closest not colliding state. Implementation of this feature is based on oriented bounding box (OBB) algorithm [10], that gives quite precise results. Current version supports only one OBB per bone. OBB's may be imported from the 3D package or defined manually.
- Key frame management: individual key frames may be browsed, modified and removed.
- Exporting the animation: this issue will be described in the next subsection.

3.3. Exporting the Motion File

Once the animation is generated it may be stored in a file in a format readable for a 3D package. KINE+ supports two formats:

- BVH: BioVision Motion Capture File format [11, 13]. It is one of the most popular MoCap (Motion Capture) file formats, imported by most 3D packages including Character Studio™ and Maya™. It contains a definition of a bone hierarchy and a set of spatial co-ordinates. Unfortunately the support given by the Character Studio™ is limited: it does not interpret data concerning individual fingers. As precise hand configuration is crucial for one of the applications for which KINE+ was originally created, an alternative file export should have been implemented.
- MS: Max Script files [12]. It is a 3D StudioMax™ native script language file. It is not usable in any other 3D package; however this file format is supported because Character Studio™ does not read finger configuration from BVH files.

3.4. Instant Renderer Unit

A small, real-time – "instant" – rendering is supported primarily for quick preview purposes [14]. Creating a full real-time 3D engine was not our objective. Currently a simple solution has been created for the Thetos system (see section 4.1, fig. 4). It uses OGL but has many limitations: among others it does not render the legs. A more advanced unit is currently developed.

4. Applications

4.1. The Thetos System: Translating Polish Written into Sign Language

The Thetos system (formerly: TGT-1) [15, 16] translates sentences written in Polish into Polish sign language. The resultant utterance is presented in form of a short, animated sequence. A simple, OGL-based rendering unit has been purposely created for the project; however the quality of the character presenting the sentences is not high (fig. 4, on the left). To obtain professionally looking animations it was necessary to export motion files to some 3D package, in this case 3D Studio-Max™ with Character Studio™. As mentioned in section 3.3, the widely used BVH file format is not fully supported by this package: hand configuration cannot be read. As it is crucial in case of sign language visualization, an alternative Max Script file has been applied to solve the problem. The resultant stills, made of animations generated by the Thetos system and rendered in 3D StudioMax™ (using available skins), are shown in fig. 4.

The animation in Thetos is generated accordingly to the description of consecutive signs made in a special text notation. This notation briefly specifies hand and finger position and orientation. This makes both FK and IK necessary. As the notation has been originally designed to be used by humans, in many cases it is inaccurate. Solving these inherent contradictions requires, among others, some collision detection and avoidance.

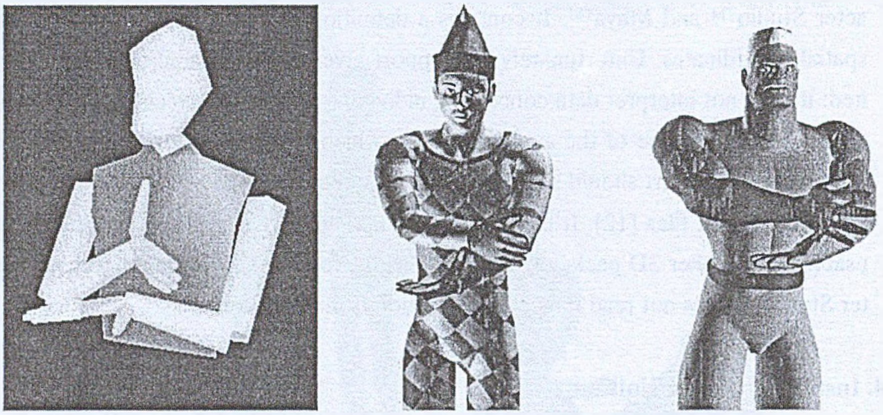
Fig. 4. Polish sign language: stills created with the Thetos system. On the left: image from an instant renderer unit. In the middle and on the right: the same sign rendered using two standard Character Studio™ skins.
Rys. 4. Polski język migowy: kadry utworzone za pomocą systemu Thetos. Po lewej: obraz natychmiastowej wizualizacji. W środku i po prawej: ten sam znak uwidoczniony za pomocą standardowych skór Character Studio™

4.2. The Auto-Animate Project: Creating Actions with Machine Learning

The Auto-Animate project, conducted in co-operation at Silesian University of technology and University of Kingston (UK), is a part of a wider FreeWill project [17]. Its general objective is to add a significant level of automation to the process of animating human and animal avatars, especially crowd scenes. A cognitive architecture has been proposed to control animation.

In the Auto-Animate project reinforcement learning algorithm has been applied to automatically generate actions for an animated avatar. It is a typical process of trials and errors. A method may be well illustrated on example of action of walking through a door. Automatic machine learning of this action requires as many as 25 million iterations; each one consists of several arm and hand rotations and strong collision detection. Number of iterations may be decreased if IK is used.

Earlier versions of the system were implemented using 3D StudioMax™ internal scripting, plug-in subsystem as well as COM-based communication. In all these cases much overhead could be observed.

The resultant action is stored in form of a BVH Motion Capture file and may be easily rendered in any 3D package (fig. 5).

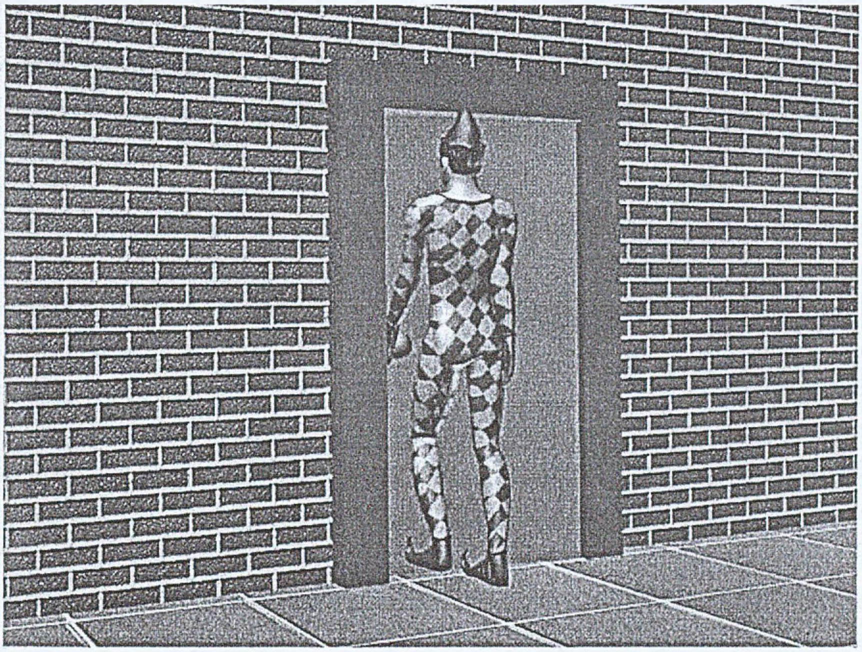

Fig. 5. Avatar going out through the door. Image created with Auto-Animate system. Rendered using a standard Character Studio™ skin

Rys. 5. Avatar wychodzący przez drzwi. Obraz utworzony w systemie Auto-Animate, uwidoczniony za pomocą standardowej skóry Character Studio™

5. Conclusion

The presented framework, KINE+, has been created to satisfy requirements of two projects realized in the Silesian University of Technology: the Thetos system [15, 16] and the Auto-Animate project [17]. The common point for both projects is that animation is generated and controlled programmatically, out of a 3D package, but it depends on models created in the package and should be exported back to the package for final rendering. The results gained are very encouraging. It seems that the KINE+ framework may be applied in any system requiring intelligent control animation.

REFERENCES

1. Maestri G.: Digital Character Animation 2. New Riders, Indianapolis 1999 (Polish edition by Helion, Gliwice 2000).
2. Giambruno M.: 3D Graphics & Animation. New Riders, Indianapolis 2002.
3. Ratner P.: Mastering 3D Animation. Allworth Press, New York 2000.
4. Badler N., Bindiganavale R., Bourne J., Allbeck J., Shi J., Palmer M.: Real time virtual humans. International Conference on Digital Media Futures. Bradford, UK, April 1999.
5. Noma T., Zhao L., Badler N.: Design of a virtual human presenter. IEEE Computer Graphics and Applications 20(4), July/August 2000, pp. 79-85.
6. Raibert M.: Human Animation and Biomechanics. Proc. of 1st Workshop on Simulation and Interaction in Virtual Environments, Univ. of Iowa, Iowa City, 1995: 215-225.
7. Tolani D., Goswami A., Badler N.: Real-time inverse kinematics techniques for anthropomorphic limbs. Graphical Models 62 (5), Sept. 2000, pp. 353-388.
8. Bindiganavale R., Granieri J., Wei S., Zhao X., Badler N.: Posture interpolation with collision avoidance. Computer Animation '94, Geneva, Switz., pp. 13-20, 1994.
9. Boulic R.: Procedural movement for articulated figure animation. Computer Graphics, Vol. 18, 1994, 453-461.
10. Gottschalk S., Lin M.C., Manocha D.: OBB-Tree: a hierarchical structure for rapid interference detection. Proc. of ACM SIGGRAPH, 1996.
11. Character Studio™ Reference and Tutorials. From Discreet. Autodesk, Inc. 2000.
12. 3D StudioMax Release 3 Reference. Vol. I and II. Autodesk, Inc. 1999.
13. BVH File Format Specification. Biovision Corp. http://www.biovision.com/bvh.html.
14. Lever N.: Real-time 3D Character Animation with Visual C++. Focal Press, Oxford 2002.
15. Suszczańska N., Szmal P., Francik J.: Translating Polish Texts into Sign Language in the TGT System. Proc. of IASTED AI 2002, Innsbruck, Austria 2002, pp. 282-287.
16. Francik J., Fabian P.: Animating Sign Language in the Real Time. Proc. of IASTED AI 2002, Innsbruck, Austria 2002, pp. 276-281.
17. Amiguet-Vercher J., Szarowicz A., Forte P.: Synchronized Multi-agent Simulations for Automated Crowd Scene Simulation. Workshop on Spatial and Temporal Reasoning with Agents Focus, International Joint Conf on Artificial Intelligence, Seattle, 2001.

Recenzent: Prof. dr hab. inż. Konrad Wojciechowski

Wpłynęło do Redakcji 30 stycznia 2003 r.

Omówienie

Animacja komputerowa obejmuje fazę modelowania obiektów trójwymiarowych, planowania ich ruchu i wreszcie końcowej wizualizacji (renderingu). Najczęściej wszystkie trzy fazy wykonuje się za pomocą specjalizowanych pakietów 3d (np. 3D StudioMax™ czy Maya™). Jednak gdy zachodzi potrzeba zaawansowanego, algorytmicznego sterowania ruchem obiektów w czasie rzeczywistym, faza planowania ruchu staje się fazą sterowania ruchem i często jest realizowana przez dedykowane moduły oprogramowania. Zaproponowana architektura, KINE+, wspomaga pozyskiwanie danych o modelu 3D z pakietu 3D, sterowanie animacją oparte na systemie kości oraz eksport utworzonych opisów ruchu z powrotem do pakietu 3D (rys. 1).

KINE+ wykorzystuje tę samą hierarchię kości, co pakiet Charakter Studio™ (rys. 2 i 3, tab. 1), co sprawia, że import danych geometrycznych z tego pakietu jest szczególnie prosty. Samo sterowanie animacją może się odbywać w oparciu o kinematykę prostą lub odwrotną. Oprogramowanie zapewnia też wykrywanie kolizji. Wygenerowane animacje mogą być eksportowane do pakietów 3D przy wykorzystaniu dwóch formatów plików.

Architektura KINE+ powstała jako odpowiedź na zapotrzebowanie stworzone przez dwa projekty realizowane na Politechnice Śląskiej. Projekt Thetos (rys. 4) to automatyczny tłumacz języka polskiego na polski język migowy. Zastosowanie KINE+ pozwoliło na stworzenie realistycznych animacji. Realizowany we współpracy z Uniwersytetem w Kingston projekt Auto-Animate obejmuje automatyczne generowanie akcji dla animowanego awatara z wykorzystaniem uczenia maszynowego. W tym przypadku zastosowanie KINE+ pozwoliło znacząco przyspieszyć czas obliczeń.

Adresy

Jarosław FRANCIK: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, francik@ps.edu.pl .

Katarzyna TRYBICKA-FRANCIK: Politechnika Śląska, Instytut Informatyki, ul. Akademicka 16, 44-101 Gliwice, Polska, kasiat@zeus.polsl.gliwice.pl .