www.cadanda.com
Computer-Aided Design AND APPLICATIONS

Generating 3D CAD Art from Human Gestures using Kinect Depth Sensor

Miri Weiss Cohen[1], Alex Frid[2], Mark Malin[3] and Vladimir Ladijenski[4]

[1]Braude College of Engineering, Israel, Miri@braude.ac.il
[2]Braude College of Engineering, Israel, Alexf@braude.ac.il
[3]Braude College of Engineering, Israel, drowqubes@gmail.com
[4]Braude College of Engineering, Israel, Vovchech83@gmail.com

ABSTRACT

In this paper, we propose a method for generating visual interactive art with 3D geometric features using Microsoft Kinect® sensor. Natural human movement and gesture recognition are used to create and interact with various objects in 3D space for art design. The Kinect output coordinates are recorded and transformed into a data structure that accurately represents the captured movement. This method of representing human gestures in a data structure provides an iterative design process that enables and preserves the sequence of human gestures either for future works or for applying transformations on existing structures. The process consists of time-dependent depth data acquisition and joint identification, followed by weighted undirected graph generation by means of a graph scanning algorithm with visual conversion and post processing. The obtained results show that various art forms can be created, ranging from 3D static designs to dynamic installations.

Keywords: Kinect depth sensor, CAD art design, human gesture recognition

1. INTRODUCTION

Microsoft Kinect was originally designed as a motion-sensing device and developed as an Xbox[14] console controller that facilitates interaction using gestures and body motion. Yet its applicability goes beyond the gaming domain. In this study, we propose a system for generating visual interactive computer-aided art with 3D geometric features using a depth acquisition sensor, such as the Microsoft Kinect sensor. Natural human movement and gesture recognition are used to create and interact with various objects in the 3D space for art design.

The research process of art design using Kinect combines three disciplines to convey a new way of creating an art design. The first discipline involves the use of body gestures to perform and create a "dance" or a language of movement. As in modern dance choreography [5], this dance conveys an "idea" that transcends from the performer to the audience. In [18] an international group of artists, programmers and dance producers aligned complex algorithmic procedures with choreographic creation and invited the audience to share in the understanding of movements and gestures. Jackson Pollock's [25] action painting or "gestural abstraction" style involves spontaneously dribbling, splashing or smearing of paint onto the canvas rather than carefully applying it. These energetic techniques depend on broad gestures directed by the artist's sense of control interacting with chance or random occurrences. The resulting work often emphasizes the physical act of painting itself as an essential aspect of the finished work. In [11] it was noted that "bodies perform images as much as they perceive external images. In this double sense, they are a living medium that transcends the capacity of their prosthetic media."

The second discipline involves the use of Kinect depth sensor [15,16,20] to capture human gestures by means of skeletal tracking, A human body is represented by a number of joints representing body parts, with each joint represented by its 3D coordinates. The goal is to determine all the 3D parameters of these joints in real time to allow fluent interactivity and thus to create a dance concept.

The third discipline is the use of graph theory to construct a weighted graph to capture the essence of the human gestures constructing the chorography of the dance.

Computer-Aided Design & Applications, 12(5), 2015, 608–616, http://dx.doi.org/10.1080/16864360.2015.1014740

Taylor & Francis
Taylor & Francis Group

Not only does this novel approach offer a system that works interactively with the human dancer, i.e., the designer, to provide straightforward output of the transformation of one media (human gestures) into another (brush strokes, dripping paints, etc.). It also provides a tool in which stages are defined and represented in data structures so that human gestures can be manipulated and transformed for future use. This approach offers a sequential art designing process, each stage of which provides a different "layer" of design that may be applied basically using the original "dance.

2. RELATED WORK

The human gesture recognition system provided by the Kinect for Windows Software Development Kit (SDK) [12,14] is widely used as an efficient tool for many fields. For example, Kinect is used in robotics and control [6,17,21,22], where a Kinect-based method allows a human operator to communicate his motions to the robot manipulators to perform tasks of picking up and placing parts in dynamic and dangerous environments. A robot equipped with Kinect can perform omnidirectional pattern recognition [23] and can measure the depth and direction of a marker quite well by using the Log-ab controller.

Filipe et al [8] proposed a system to assist blind people in navigating the world, thus replacing the white cane. This system provides information about the surrounding environment in real time, such as no obstacles or obstacle ahead and reports on different planes, such as stairs. Other uses include facial feature recognition with human–computer interaction for cognitive rehabilitation [9] and facial expression tracking and representation by avatars [24].

Groentjes [10] uses Kinect with holography, proposing a dynamic holographic image that can be manipulated using gestures and captured movements. Le et al. [15] used the skeletal information provided by the SDK for human posture recognition in the context of health monitoring. They developed a joint database using Microsoft's Skeletal API with pre-recorded postures for matching the captured posture with the images in the database.

Virtual character armatures were interactively controlled in real-time by tracking body poses [19] using their own body poses and not the key framing approach, thus yielding realistic and smooth animations. The INTEGRARTE project [2] provides a body experience through movement visualizations and sounds to create brushstrokes, circles and echoes. Their system provides interactive real time artwork that offers the participant a complete artistic experience but cannot be saved or manipulated

Chattopadhyay et al. [3] studied human-motion-initiated music generation. Each joint was assigned its own customizable sound source to create musical notes in real time. Body movements in were defined by velocity, acceleration and position change and every movement generated musical notes with respect to its features.

3. PROPOSED APPROACH

The proposed approach is based on the notion of transforming a set of movements constrained by time to create a 3D art form that translates one visual media (such as natural human movements) into another (for example: illustration or drawing). This transformation process, depicted in Fig. 1, consists of the following stages: (i) data acquisition and windowing, (ii) data stream to graph transformation, (iii) graph traversal, (iv) visual transformation, and (v) post processing.

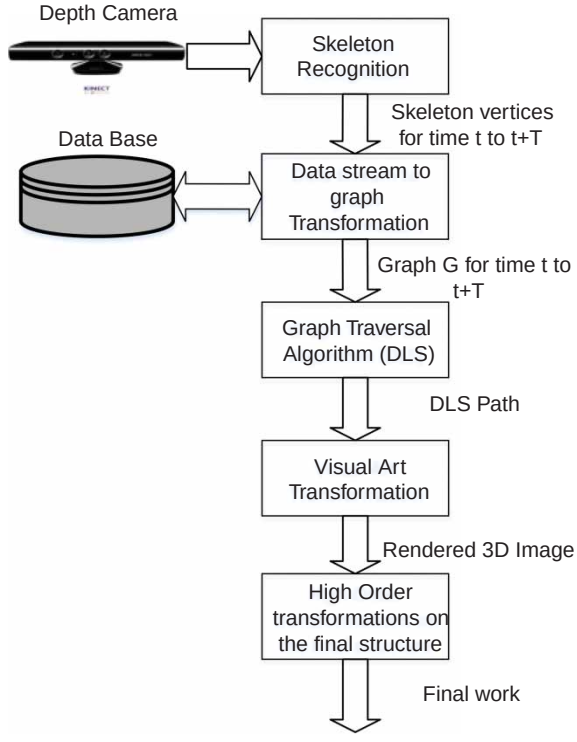

Fig. 1: The process of translating human gestures to CAD art.

3.1. 3D Sensor Data Acquisition

The Kinect sensor was introduced in November 2010 by Microsoft for the Xbox 360 video game system. The device was designed to be positioned above or below a video display and to track player body and hand movements in 3D space [14,15], thus allowing a new type of interaction with the Microsoft game console using natural body movements (i.e., human gestures). The Kinect sensor contains a color sensor (an RGB camera), a depth sensor (consisting of an IR light source emitter and IR depth sensor), and a tilt

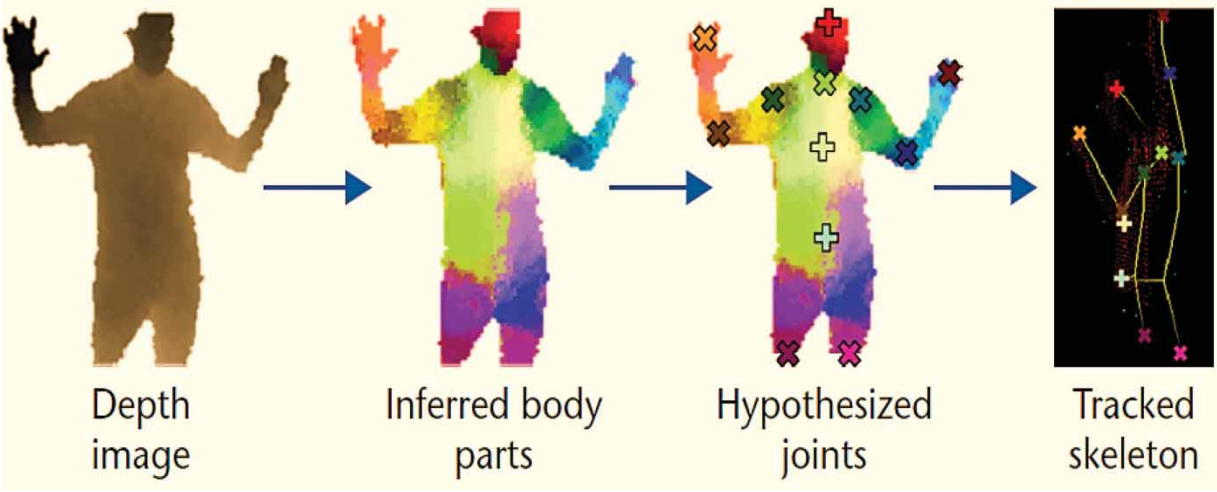

Fig. 2: Pipeline of skeleton data acquisition stage for a single image [24].

motor. In this study we used only the depth sensor abilities in order to acquire the human body parts (skeleton) stream.

Figure 2 describes the process of skeletal data acquisition, which consists of two stages. The first stage involves depth space data acquisition (first image in Fig. 2). At each frame, the depth sensor captures a gray scale image of everything visible in the sensor view field, where each pixel contains the Cartesian distance from the camera plane to the nearest object. The second stage comprises the skeleton space (see last three images in Fig. 2). Each frame captured by the depth sensor is processed by the Kinect runtime into skeleton data containing the 3D position of a skeleton and each of the skeleton joints (stored as x, y, z coordinates). This conversion to skeletal space from depth images is accomplished by means of supervised learning, which requires a large amount of marker-based motion-capture data for training. Thus the detection pipeline involves three major steps. In the first step (inferred body in Fig. 2), the closest known body parts are roughly estimated. In the second step, the joints are estimated based on the previous step of body parts detection (hypothesized joints in Fig. 2). In the last step (tracked skeleton in Fig. 2), the marked skeletal parts from the previous step are fine-tuned using physiological data knowledge regarding the location of the human joints and then joined into a single skeleton.

3.2. Depth Data to Graph Transformation

The motivation for representing the data in a graph is twofold: first, to maintain and capture the essence of the set of movements provided by human gestures, and second, to save the set of movements for future work. Representing the set of movements over a time span will result in a complex graph and will enable several search methodologies to translate it to an art form. This graph representation provides the ability to apply transformations on the original graph, and creating different designs (applying different geometrical features, brushes etc.,) for a fixed set of movements, as detailed in the following section.

Given 20 possible joints of the human skeleton using the Kinect's SDK as input, we define a graph in the following form. Let G= (V,E) be a graph with n vertices, such that $V = \{v_i | v_i \in R^3, i = 1, \ldots, n\}$ denotes the set of vertices (joints) and $E = \{e_{ij} = (v_i, v_j | v_i, v_j \in V,\ i \neq j\}$ represents the edges. Each joint is represented by (x_i, y_i, z_i, t_i), where *x,y,z* are the location coordinates of the joint and t is the time that indicates the human gesture sequence. At each joint (x_i, y_i, z_i) at time $t = 0$, a weight function w' is valued randomly between *1.. i. w'* represents one of the predefined shapes that we chose. Each graph traversal (up to number of shapes) sums those values to decide upon the shape and is assigned once per design. The value is decided randomly in order to create variety and eliminate preferable of any shape in the design creationWe calculate the following two weight functions *w1* and *w2*, to serve as weights on the graph edges, providing additional information regarding total and relative distances.

$$w_1 = \sqrt{\Delta x^2 + \Delta y^2} \tag{3.1}$$

$$w_2 = (x_i - x_{i+1}, y_i - y_{i+1}) \tag{3.2}$$

These values will be used to evaluate the human gestures in the following stage.

The Kinect device produces information in sequential order of 30 fps, and the number of joints being tracked ranges between 10 and 20. This yields a complex problem of representing an enormous number of joint coordinate values over a time span. The options for feasibly tracking the joints are as follows: reducing the number of joints to be tracked by the Kinect, or reducing the number of frames per second.

In our study we chose to reduce the number of joints to 13 out of a possible 20. We found this most beneficial for tracking the movement of skeletal joints coordinates when extracted from a recorded video sample. The joints coordinates were expressed in meters in space [20] and were converted to millimeters. During the time spans of human movement, the data is stored in an array of skeletal joints, along with the weight function values (w_1, w_2^x, w_2^y), calculated by Eqn. (3.1) and Eqn. (3.2). Tab. 1 shows an example of the raw and calculated data.

Thirty frames are added to the raw data file each second. Each frame contains all 13 joints along with the relevant coordinates and calculated weights. Each frame represents a different depth layer, resulting in 30 different layers of depth. We calculate the following equation:

$$joint\_number + (13 * \mathrm{d}) \tag{3.3}$$

where d denotes the depth factor, thus providing a sequence of indexes for the resulting depth factor of each joint. In Tab. 1, each row indicates a time stamp, and every six columns, starting from the second column, indicate the weight function values (w_1, w_2^x, w_2^y) and the axis *(x, y and z)* of what is depicted in the first row above the joint. The table depicts only an example of two joints, the head and the shoulder center. The weight function W_1 is the direction vector, while $W_2.^x$ and $W_2.^y$ coincides the distance and is time dependent. Translating the above to motion measurements yields, w_1 is the parameter controlling the direction and movement of each shape. W_2, ($W_2.^x$ and $W_2.^y$) is a position measurement of each limb or body part along the time axis. Based on the data in Tab 1, we calculate the distances and change in direction of each one of the joints.

Figure 3 is an illustration of the graph constructed from the data in Tab. 1, where the nodes represent the joints and the edges are weighted by both Eqn. (3.1) and Eqn. (3.2).

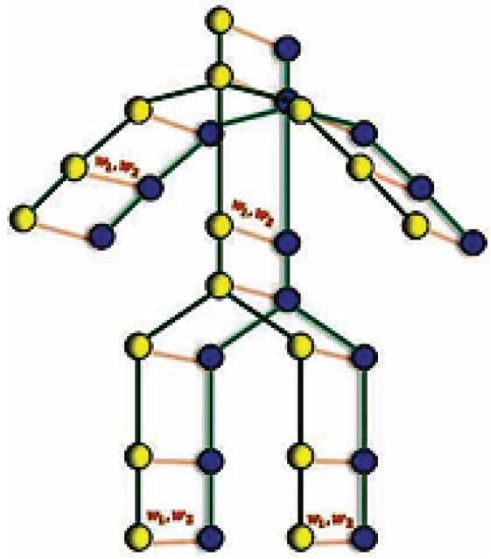

Fig. 3: Illustration of the graph G(V,E).

The graph representation tends to grow over time. Moreover, traversing the graph and retrieving information is based on the definition of the data structure.

An adjacency matrix was chosen to represent the graph, due to query time complexity of $O(1)$. Fig. 4 is an illustration of how an adjacency unfolded matrix is defined. The basic idea is to record and save all body joints coordinates and multiply them by the depth factor 30 times in each timestamp. This process results a huge matrix 13*30*t in size. For example, a 10 second timestamp, results a matrix of size *(13*[30*10])*(13*[30*10])= 3900 × 3900.*

3.3. Graph Transformation to Visual Features

The graph constructed in the previous stage serves as base for the art design and requires that the

Time	X	Y	Z	W_1	$W_2.^x$	$W_2.^y$
			Head			
12:54:17:311	3.136883	62.08597	158.1259	7.336379	−0.73252	0.0404
12:54:17:357	2.404358	62.12637	157.8679	24.86858	−2.46644	0.318021
12:54:17:373	−0.06208	62.44439	156.9579	48.90922	−4.89018	0.085044
12:54:17:420	−4.95227	62.52943	153.1063	11.32383	0.919348	0.661129
12:54:17:435	−4.03292	63.19056	153.6269	26.62431	−2.46425	1.007968
12:54:17:482	−6.49717	64.19853	153.4303	57.06237	−5.68975	−0.43343
		Shoulder	Center			
12:54:17:311	−4.54363	45.39131	163.6053	8.298117	−0.81381	0.162187
12:54:17:357	−5.35744	45.55349	163.1052	25.23027	−2.413	0.736946
12:54:17:373	−7.77044	46.29044	161.5696	16.37976	−1.54939	0.531375
12:54:17:420	−9.31983	46.82182	160.3314	24.18519	−2.31929	0.685653
12:54:17:435	−11.6391	47.50747	158.7537	21.41993	−2.12138	0.296429
12:54:17:482	−13.7605	47.8039	157.2459	20.32721	−2.02824	−0.13494

Tab. 1: Example of raw data (head and shoulder).

colors and shapes be defined. The graph (saved in the system) provides the ability to manipulate using various transformations. Art design can be influenced by style and artistry if expressed by a combination of transformations.

3.3.1. *Depth Limited Search algorithm (DLS)*

The DLS algorithm is used in our approach for retrieving the relevant information from the graph constructed in Section 3.2. Although DLS is a search algorithm used for traversing the graph, we use different versions of the DLS for each weight function: w_1, w_2, w'. The most general DLS algorithm is used for traversing the weight function w_2 and is explained in detail as follows:

1. Choose a random joint from the array by drawing an index. The joint can be any joint at any depth, *Joint*$\{i, d\}$, where i = the sequence number of the joint $(1, \ldots, 13)$ and d = the depth.
2. Choose each of the adjacent joints from the adjacency matrix, and place those joint values in a temporary stack. Each joint has at least one adjacent joint for assuring the search progress. Moreover, the selected joint has values for different depths $d+1$ and $d-1$.
3. Choose one joint from the stack group to continue.
4. If the search reaches the predefined limit, which is the number of passes from joint to joint, then stop and return a *Point3D* object, which holds three final weight function values for each axis x, y and z. Otherwise, repeat from step 2. Only traversing from edges between depth layers, from joint at d to joint at $d+1$ or $d-1$, contributes to the DLS search algorithm limit count.

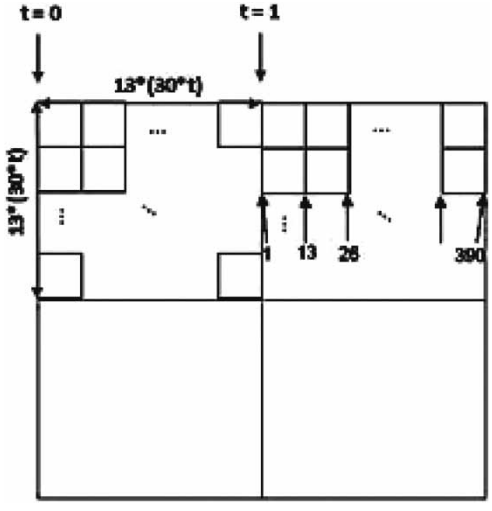

Fig. 4: Adjacency unfolded matrix.

The version of the DLS algorithm for weight function w_1 retrieves w_1 and w_1^1, w_1^2 and w_1^3 during the same single run as the weight functions calculated for each axis x, y and z separately. This is achieved by dividing the DLS run into three steps, constrained by the predefined limit of the search. Each part sums distinct w_1 values.

For example, if the DLS limit is twelve, the search algorithm must traverse precisely twelve edges until it stops. By dividing the same DLS run into three steps, the algorithm will sum the w_1^1 result after traversing the first four edges as the first step. The second sum w_1^2 will result after traversing the next four edges as the second step, and the last sum w_1^3 will result after traversing the last four edges as the third and last step. Hence, one complete run will produce three distinct values w_1^1, w_1^2 and w_1^3 of the weight function. The weight function w' is calculated and retrieved in the first depth layer.

The DLS algorithm was chosen because it achieves sufficient results in two ways. First, even when a specific starting joint is chosen (random process), the search algorithm prevents repeatability. Second, run time is minimal, resulting in efficient response time.

3.3.2. *Translation to art design features*

Given the graph as input, translation to art design features consists of the following:

1. Defining shapes and colors

Five basic shapes were chosen for the design: pyramid, cube, cylinder, sphere and cone. These particular shapes were chosen because they are suitable for representing geometric abstract art and can be easily used with CSG or B-rep for combined shapes. The shapes were created using the 3D graphics [1,4] infrastructure based on DirectX [16,7] functionality, which WPF (Windows Presentation Foundation) supports.

The choice of the number of 3D shapes and of the background is predefined by the user, as depicted in Fig. 5. The user sets the time during which the system represents the human gestures. We have found this useful for achieving a specific design of style. Choosing a random number of shapes is straightforward, but experience using the system indicates that users prefer the predefined option.

For each shape, a joint is chosen for defining the starting node for the DLS algorithm. The starting joint will also define the initial color of the shape. The color is defined by the joint's coordinates (x_i, y_i, z_i) to the corresponding RGB respectively. Each search will sum the weight w' on the edges, which corresponds to a predefined shape.

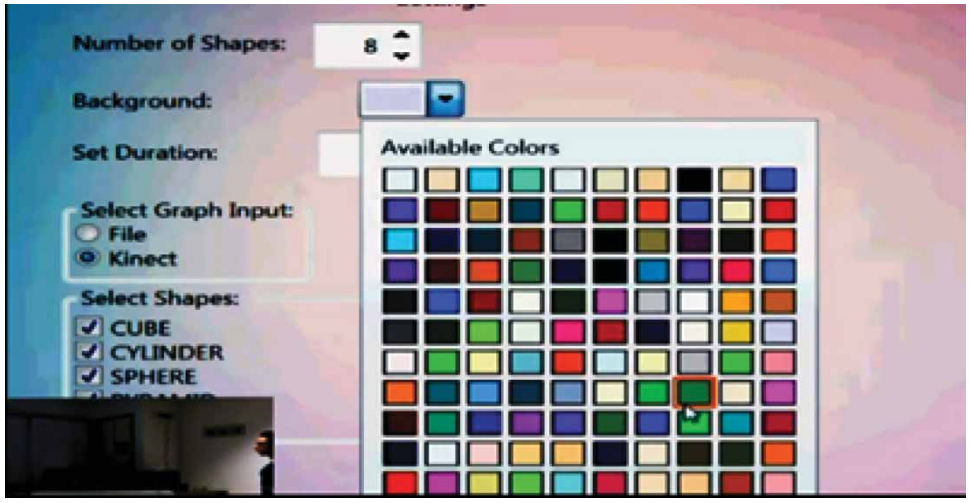

Fig. 5: Snapshot of the system with options defined by the designer.

2. Defining the movement and the scaling of the shapes:
   a. Randomly choose a joint to start the DLS algorithm, to result in the value of w_2.
   b. The movement of each shape will be defined by direction vector w_2, which also defines the speed.
   c. Choose another joint randomly to start the DLS algorithm, which calculates w_1.
   d. Each shape will be transformed by applying the following scale matrix:

$$s = \begin{bmatrix} w_1^{[i]} & 0 & 0 \\ 0 & w_1^{[i+1]} & 0 \\ 0 & 0 & w_1^{[i+2]} \end{bmatrix} \quad \text{where i is the edge number} \tag{3.4}$$

The final stage is the output of a 3D design. In this stage, after the graph and the visual transformation are applied, an art design will be created. This design can be saved along with the graph to allow further interaction and modification of the design.

4. RESULTS

The software developed for 3D art design is in its first stage of development, and therefore basic shapes have been implemented. Any OpenGL complex figures can be added to the software and used for creating numerous compositions. Each of the human gesture movements is saved in the system as a graph capturing the composite human gesture we refer to as "dance" and serves as the basis for several different art forms for a certain dance. Each dance may be expressed as different output, producing various art designs for each dance.

The process of transforming a dance into an art design is not interactive. The human performs a set of human gestures. These are saved in a graph, and the output is the result of the translation process.

Figure 6 shows a snapshot of a single moment of a dance (a), with the parts of the joints depicted on the human body by red circles. The edges are depicted by red lines (b). As the human body is a rigid body, the connections between joints do not change over time.

Figure 7 depicts four snapshots of an art design produced from eight basic shapes from a human gesture dance of 6 seconds. The first composition art design output, depicted in 7(a), is the basic translation of the graph constructed from the dance. Fig. 7(b-d) shows the designs created by manipulating the graph using the human gestures again to apply transformations on the first basic graph.

Figure 8 is an attempt to transform *Thirteen-Rectangles-1930* [13], an existing 2D abstract art piece by Wassily Kandinsky which serves as the inspiration for a 3D interactive art design. Fig. 8(b) shows the original work, while Fig. 8(a) is the output of the dance performed (by the user) to capture the essence and composition for translating from one media to another. In this example the artistic work is done from the art design to the human gestures made for creating it. Our system facilitated the ability to use an existing art work and convert it to a 3D dynamic art design, manipulated and performed by human gestures. Moreover, the system enabled designing a new art composition based on the dance and transformation implementations.

Figure 9(a) is an example of a static art design translation for human gestures, and Fig. 9(b) is an illustration of a possible installation (3D defined art design in 3D space) for dynamic 3D art design using our system. This option of art designs could be done and performed by a group of humans, producing a mixture of dances and designs

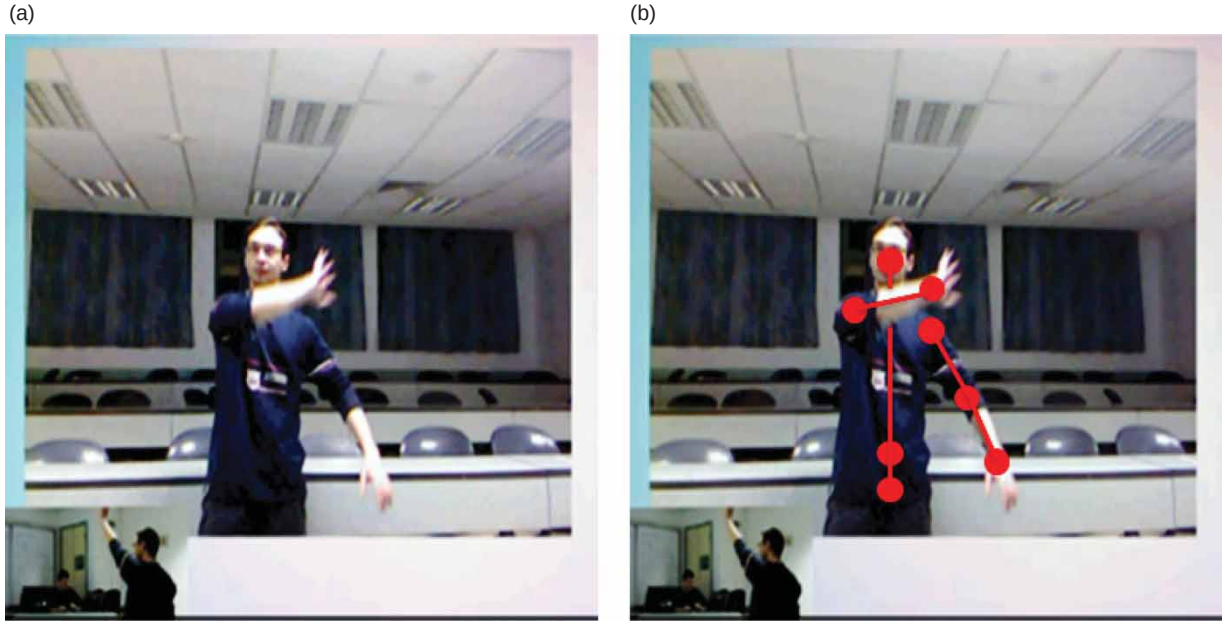

Fig. 6: Human gesture: (a) joints depicted by red circles and (b) edges depicted by red lines.

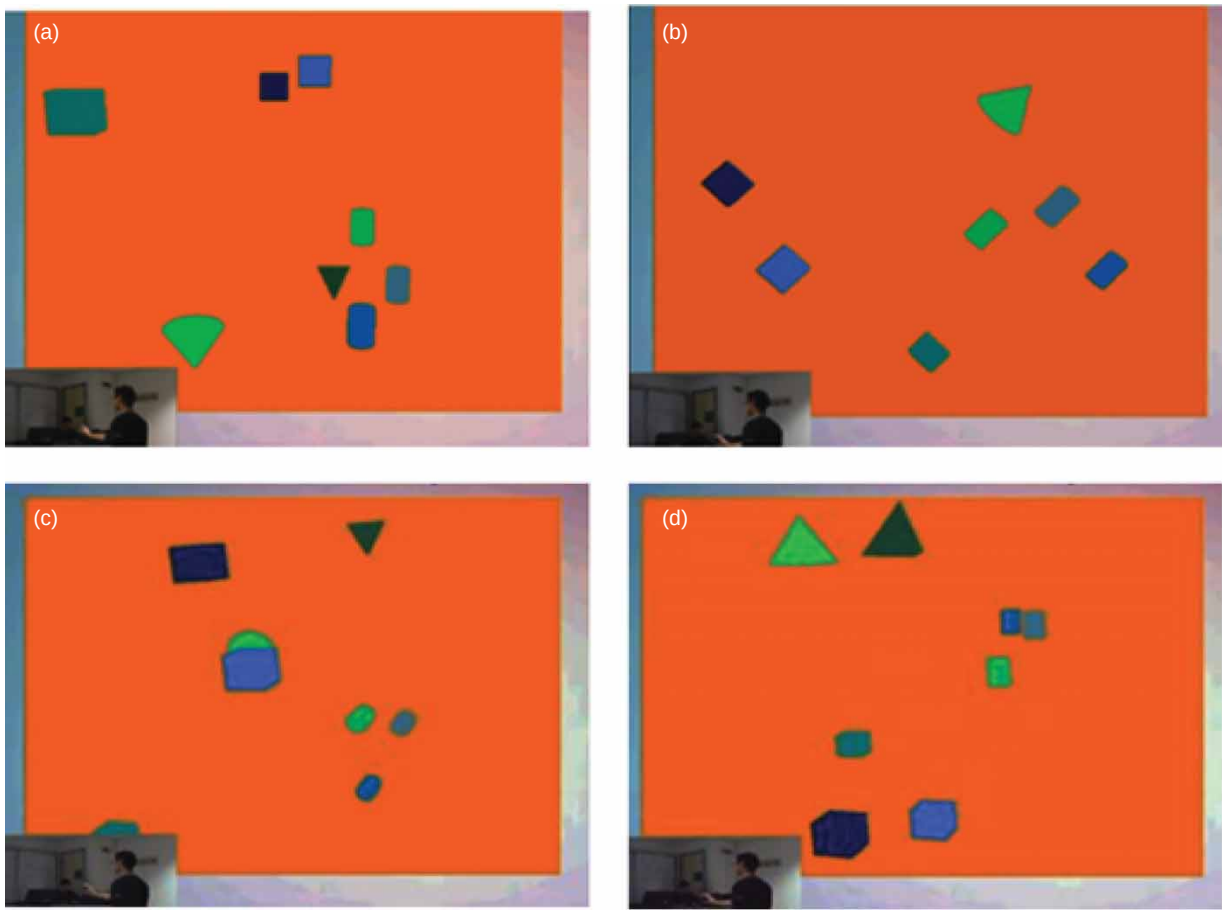

Fig. 7: (a) basic art design; (b-d) 3D transformations made by hand gestures.

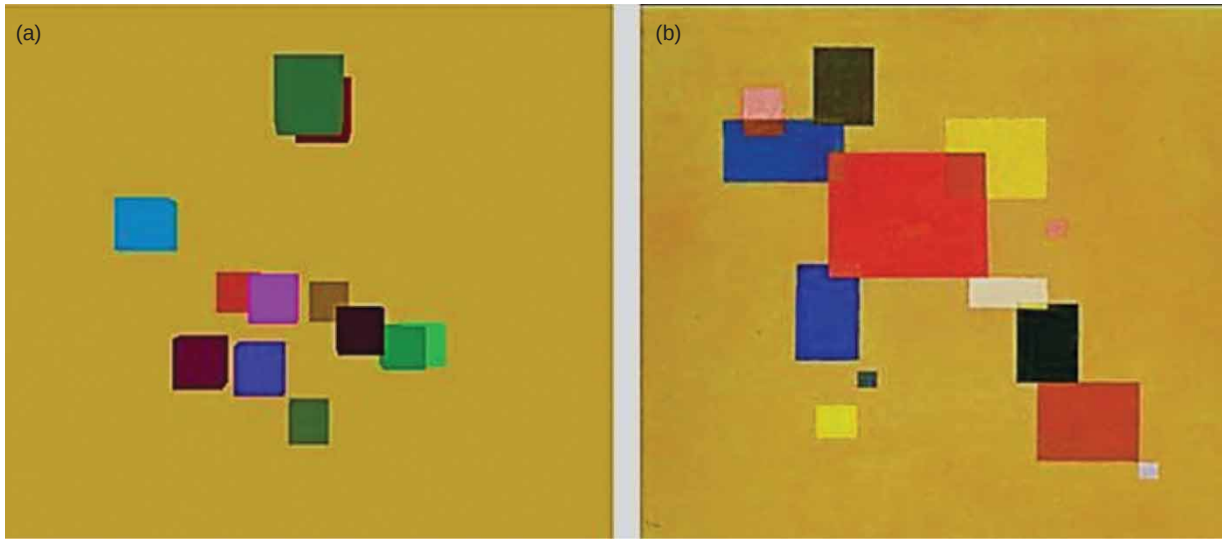

Fig. 8: (a) Our system art design snapshot, (b) original piece by Wassily-Kandinsky.

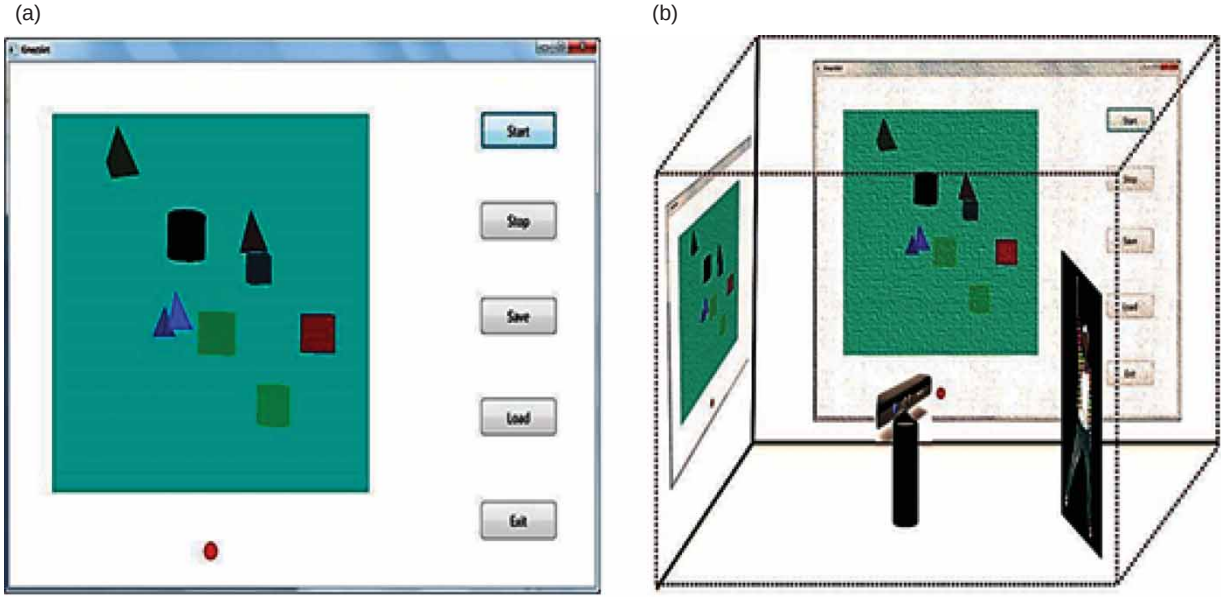

Fig. 9: (a) Snapshot of an art design; (b) installation of 3D dynamic art design.

5. CONCLUSIONS

The 3D art design system proposed in this paper generates an abstract and unique geometrical design based on natural human gestures and movements. Shapes, colors, placements and speed are defined, captured and translated into art forms. Because each individual is unique and each movement is different and cannot be replicated precisely, the process will yield a new unique design each time and save the combination of motions in a weighted graph. This process of representing human gestures in a data structure provides an iterative design process. Interestingly, though the design process is fully computerized, the generated structures are personal, natural and very similar to those of humans.

REFERENCES

[1] 3D Graphics, http://msdn.microsoft.com/en-us/library/ms742562.aspx, Microsoft, 2013.

[2] Castro, B.-P.; Velho, L.; Kosminsky, D.: Integrate: digital art using body interaction, Computational Aesthetics in Graphics, Visualization and Imaging, Cunningham D. and House D. (Editors), 2012.

[3] Chattopadhyay, D.; Vallier, T.; Berg, T.; Schedel, M.: Interactive Music: Human Motion Initiated

Music Generation Using Skeletal Tracking By Kinect, November 2011.
[4] Coordinate Spaces, http://msdn.microsoft.com/en-us/library/hh973078.aspx, Microsoft, 2013.
[5] DeLahunta, S.; Bevilacqua, F.: Sharing descriptions of movement, International Journal of Performance Arts & Digital Media, 3(1), 2007, 3–16. http://dx.doi.org/10.1386/padm.3.1.3_1
[6] Du, G.; Zhang, P.: Markerless human–robot interface for dual robot manipulators using Kinect sensor, Robotics and Computer-Integrated Manufacturing 30, 2014, 150–159. http://dx.doi.org/10.1016/j.rcim.2013.09.003
[7] Extended WPF Toolkit, https://wpftoolkit.codeplex.com/, Xceed, Aug, 2013.
[8] Filipe, V.; Fernandes, F.; Fernandes, H.; Sousa, A.; Paredes, H.; Barroso, J.: Blind navigation support system based on Microsoft Kinect, Procedia Computer Science, 14, 2012, 94–101. http://dx.doi.org/10.1016/j.procs.2012.10.011
[9] González-Ortega, D.; Díaz-Pernas, F.J.; Martínez-Zarzuela, M.; Antón-Rodrígue, M.: A Kinect-based system for cognitive rehabilitation exercises monitoring, Computer Methods and Programs in Biomedicine, 113, 2014, 620–631. http://dx.doi.org/10.1016/j.cmpb.2013.10.014
[10] Groentjes, T.: Holography and Kinect, Leiden University, January, 2013. http://www.liacs.nl/assets/ . . . /2012-2013-02-TomGroentjes.pdf.
[11] Hans. B.; Image, medium, body: A new approach to iconology. Critical Inquiry vol. 31,(2005) The University of Chicago Press. http://arthistory.about.com/od/glossary_a/a/a_action_painting.htm
[12] Jana, A.: Kinect for Windows SDK Programming Guide, Packet Publishing, Birmingham, UK, December, 2012. ISBN: 1849692386.
[13] Kandinsky, W.: Thirteen-Rectangles-1930, Musée des Beaux Arts, Nantes, France.
[14] Kinect for Windows features, http://www.microsoft.com/en-us/kinectforwindows/discover/features.aspx, Microsoft, 2013.
[15] Le, T.-L.; Nguyen, M.; Nguyen, T.: Human posture recognition using human skeleton provided by Kinect, 2013 International Conference on Computing, Management and Telecommunications (ComManTel), January 21–24, 2013, 340–345. http://dx.doi.org/10.1109/ComManTel.2013.6482417
[16] Microsoft, Kinect4Windows Developers Website (http://www.microsoft.com/en-us/kinectforwindows/) 2013.
[17] Oliver, A.; Kang, S.; Wonsche, B.C.; MacDonald, B.: Using the Kinect as a Navigation Sensor for Mobile Robotics, Proceedings of the 27th International Image and Vision Computing New Zealand Conference (IVCNZ 2012), 26–28 November 2012, Dunedin, New Zealand, pp. 509–514.
[18] Rodrigues, D.-G; Grenader, G.: MotionDraw: A Tool for Enhancing Art and Performance Using Kinect, in CHI '13 Extended Abstracts on Human Factors in Computing Systems, NY, USA, 2013, 1197–1202. http://dx.doi.org/10.1145/2468356.2468570
[19] Sanna, A.; Lamberti, F.; Paravati, G.; Rocha, F. – D.: A Kinect-based interface to animate virtual characters, J. of Multimodal User Interfaces, 7, 2013, 269–279. http://dx.doi.org/10.1007/s12193-012-0113-9
[20] Sensor setup and support, http://www.microsoft.com/en-us/kinectforwindows/purchase/sensor_setup.aspx, Microsoft, 2013.
[21] Stowers, J.; Bainbridge-Smith, A.; Hayes, M.: Quadrotor Helicopter Flight Control Using Hough Transform and Depth Map from a Microsoft Kinect Sensor, IAPR Conference on Machine Vision Applications, June 13–15, 2011, Japan.
[22] Susperregi, L.; Arruti, A.; Jauregi, E.; Sierra, B.; Martínez-Otzeta, J.M.; Lazkano, E.; Ansuategui, A.: Fusing multiple image transformations and a thermal sensor with kinect to improve person detection ability, Engineering Applications of Artificial Intelligence 26, 2013, 1980–1991. http://dx.doi.org/10.1016/j.engappai.2013.04.013
[23] Tsai, Z-R.: Robust Kinect-based guidance and positioning of a multidirectional robot by Logab recognition, Expert Systems with Applications 41, 2014, 1271–1282. http://dx.doi.org/10.1016/j.eswa.2013.08.025
[24] Zhang, Z.: Microsoft Kinect Sensor and Its Effect, IEEE Multi Media, 19(2), 2011, 4–10. http://dx.doi.org/10.1109/MMUL.2012.24
[25] http://arthistory.about.com/od/glossary_a/a/a_action_painting.htm