3D Geology Scene Exploring Base on Hand-Track Somatic Interaction

Wei Zhang[1], Fang Zhu[2], Ping Lu[2], Ping Li[3], Bin Sheng[1], and Lijuan Mao[4]

[1] Shanghai Jiao Tong University, Shanghai, People's Republic of China
shengbin@sjtu.edu.cn
[2] ZTE Corporarion, Nanjing, People's Republic of China
[3] The Hong Kong Polytechnic University, Hong Kong, People's Republic of China
[4] Shanghai University of Sport, Shanghai, People's Republic of China
maolijuan@sus.edu.cn

Abstract. Terrain analysis is the basis of geological research. However, due to factors such as distance and range, it is often difficult to study the terrain environment in the field. Therefore, researchers can observe and study the terrain by making a three-dimensional terrain model. The 3D terrain model can reduce the terrain range, eliminate the limitation on distance, and control the scene through program interface, to achieve human-computer interaction to meet different research needs. The usual human-computer interaction methods are implemented through traditional peripherals such as the mouse and keyboard. With the rapid development of computer network technology and the continuous improvement of intelligent software and hardware, people have greater requirements for interactive manipulation and immersion. This article proposes a method for displaying terrain models based on real-sensing technology by using Intel's RealSense camera to control the scene of the desert model through gestures. The user can observe the model from two different perspectives, and use different gestures to zoom in, zoom out, move, and rotate the scene, as well as choose some options. The traditional method of controlling by mouse is also applicable. The entire project is designed as a game, with a realistic and complete model, an exquisite interface, and strong interactivity.

Keywords: Somatosensory · 3D Scene · Gesture sensing · Unity · Hand track · Depth camera · RealSense

1 Introduction

Motion-sensing technology has become one of the most popular methods for human-computer interaction. Besides mouse and multi-touch, somatosensory interaction may lead to the third revolution of human-computer interaction. With the passage of time, many famous hardware corporations have released their own commercial visually somatic peripheral equipment, including Microsoft's Kinect, Intel's depth camera and so on. Even though there have been numerous advancements in the field of AR and VR, it is still an emerging field. With all these factors, the loss of suitable interaction method is quite a problem.

In somatic interaction field, we can track eyes, body movements, facial expressions, gestures etc. Eyes tracking is a new technology and so far under experimentation. Body movement based on gyroscope is the first generation of somatic method, which has later been applied in many game industries, for example, Nintendo's somatic game console Wii. However, a gyroscope is not suitable to generate reliable digital data which can be used to model accurate body skeleton information, and then simulate available input data. Visual body movements capture technique based on depth camera is now another method to detect body language. Microsoft's Kinect has been in production with their game console XBOX for years. But the high price keeps it away from mainstream markets. In comparison to this, for movements capture, Kinect's error rate and reaction time has quite a distance from traditional input method like mouse or keyboard. In this paper we applied Intel's new sensor device, the RealSense SR300 depth camera, in our 3D scene exploring platform. Compared with Kinect, SR300's price level is much more reasonable, and makes it more expansible and more developable.

To fulfill gesture interaction, Kinect released a set of official units. For our project, we need to analyze the original image data generated by depth camera, then design recognition algorithms to sort and process them and finally transmit the signal to game controlling input APIs. In addition to this, we also built a 3D game platform to test our techniques using the Unity engine.

Compared with other popular commercial depth cameras such as Kinect, the SR300 is a much cheaper choice, which means that depth camera can be applied in much more fields. And due to Intel's SDKs and development support, SR300 is more open to other developers, and is more friendly to embedded hardware platforms such as Raspberry Pi, etc.

2 Related Work

2.1 Image-Forming Principles of Depth Camera

Depth camera is a kind of camera which applies reflection-based methods to acquire not only color and position of objects in the camera view, but also their distance to the camera. There are three kinds of image-forming principles in the market at present, namely structured-light, stereo vision and time of flight. The RealSense applies the structured-light principle, which casts an encoded line light source to the objects and demodulates their distance from the structural light pattern. Leap Motion applied stereo vision principle, which captures two images of the same scene with two cameras in different relative positions just like what the human eyes do. Then the system will use parallax to calculate the distance of the objects. The RealSense SR300 used in our project applied time of flight principle, which will be discussed in detail in Sect. 3.1.

2.2 Gesture Recognition

The most commonly used human gesture recognition methods are depth-based and skeleton-based [15]. Depth-based methods recognize gestures by filtering out the noise of input and obtaining global features. Existing work mainly differs in how to generate representative global features. For example, HON4D [9] uses a 4D surface normal orientation histogram to describe the depth sequence. The improved HON4D [17] forms a poly-normal, used to characterize local motion and shape information by clustering hyper surface normal vectors in the depth sequence. Skeleton-based methods can be divided into joint-based and body part based. Joint-based methods use a coordinate system to relatively model the position and motion of a couple of joints. One way is to relatively combine the position of different joints to form a coordinate system, like in [6,13]. On the other hand, the human gesture can be recognized by calculating joint orientations against a fixed coordinate system, like in [16]. While in body part based methods, human body parts are represented by some rigid cylinders connected by joints to form an articulated system. Gestures can be distinguished through the 3D geometric relationships between joint cylinders [4,14,18]. After this, the data will go through either handcrafted feature extraction and feature representation or deep learning methods to extract human gesture descriptors. As for Kinect V2 sensor, it evaluates each pixel of the depth image and distinguishes human body parts from the environment background. This process combines a few computer graphics and vision techniques such as edge detection, noise threshold processing and categorization of human body features. Kinect V2 sensor can actively track two players' skeletons and create segmentation masks for each tracked player to reduce calculation. Then the new depth image which has culled background objects is transmitted to Exemplar, a machine learning system that can recognize specific body parts. The last step is to generate a skeleton system according to 25 tracked joints, so that game developers can select and combine preferred components to create unique game experience.

2.3 Natural Interaction Interface

Natural human-computer interaction (HCI) system is an interactive framework that integrates human behavior into technological applications [12]. DesertBox is supposed to be designed as one of such interfaces. Natural interaction systems consist of several modules, such as sensing subsystem and presentation module. The sensing subsystem gathers data from different dimensions in the surroundings with specific sensors, including visual sense, auditory sense, tactile sense, etc. The presentation module is responsible for integrating the signals from sensors and generating output results. There are some rules that should be kept in mind when designing an HCI system. First of all, digital elements in the interface should behave like their counterparts in the real world to give users an intuitive experience. Secondly, the design of the interface should be lightweight and minimalist to help users pay attention to the rich content of the application instead of distracting them with additional operating instructions.

HCI systems have various application scenarios, such interaction with realtime virtual actions [7,8], as multimedia browsing which allows the user to explore multimedia objects intuitively, knowledge building which allows multiple users to accomplish tasks collaboratively [2], interactive exhibition where visitors can enjoy multi-sensory experimental experience [1] and interactive system which stimulates for learning [5,11].

3 Gesture Recognition

3.1 Hardware System

We used Intel's RealSense SR300 to get the original gesture depth image. The RealSense SR300 is a real time sensor camera produced by Intel. Supported by Intel's Realsense SDKs, it can realize gesture sensing with convenient APIs. Combining depth measuring and a 1080p RGB color camera, the application will detect the shape of a moving hand and then model the hand skeleton, realizing Skeleton Tracing.

3.2 Recognition Algorithm

Fig. 1 shows the core algorithm structure of action recognition. We divided the gestures into 2 types of gestures: static and dynamic. The dynamic gestures can also be constituted with several static ones [10].

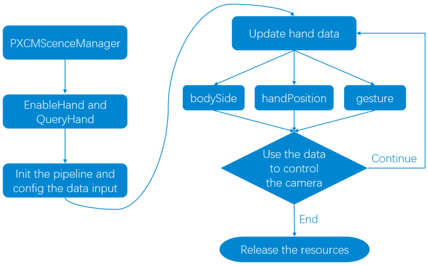


Fig. 1. Core algorithm structure of action recognition

Traditional gesture recognition [19] compares the density of depth data and body movement model to analyze the components of the human body. It then, uses inverse dynamics to reflect this data back to body movement model to realize gesture recognition.

Jamie Shotton [20] proposed a method to measure hand joints through single depth graph without time data. For the pixel x in depth graph I, its eigenvalue is:

$$f_{\theta}\left(I,x\right)= \ d_I(x \ + \ \frac{u}{d_I(x)})- \ d_I(x \ + \ \frac{v}{d_I(x)}) \tag{1}$$

In the expression: $d_I(x)$ is the depth of pixel x. When a user enters the recognition area, no matter how far they are from the camera, a coordinate system can always be set on their body. $\theta(u, v)$ is used to confirm the pixel's offset in this system with u and v. Multiply the normalized $d_I(x)$ with these two offsets, then we can get two depth values. Their difference is the eigenvalue f_θ. As for background pixels, their $d_I(x)$ values will be a vastly positive constant. Reaching body structure tags, we can use decision-making tree to sort the pixels in a graph and tag them. Suppose a decision-making forest with size T has a tree t. Every leaf node in this tree contains an eigenvalue f_θ and a threshold τ. In the tree t, there is a known distribution function $P_\tau(C|I, x)$ with the body structure tag C. This function is used to describe the distribution of pixels as:

$$\omega_{ic} = P(C|I, x) \cdot d_I^2(x) \tag{2}$$

In the expression: ω_{ic} is the weight of the pixel, which is used to describe the probability of the tag that fits this pixel. There are several embedded gestures in Intel's RealSense SDK (Fig. 2). Using these embedded gestures, we constructed new gestures to realize game function.

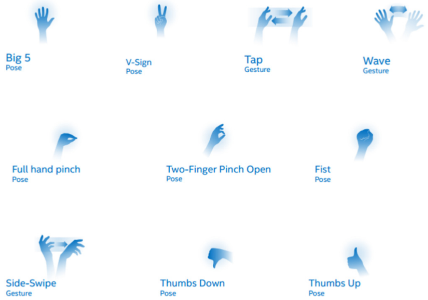


Fig. 2. Samples of intel RealSense SDK embedded gestures

3.3 Shaking Optimization Algorithm

During our recognition progress, due to slight shaking of hands, some operations, especially rotation, will cause an obvious unsteady viewpoint. To solve this, we designed an optimization algorithm 1. This procedure will construct a queue with a size of five, to contain hand skeleton position data. When the depth camera loses the position of hands, it clears the original queue, and fills it with new information, to avoid data from the shaking position, after it re-captures the hands. Every time the position data is requested, the algorithm returns the arithmetic mean value of queue.

Algorithm 1. Shaking Optimization

```
input(original single hand position data)
output(optimized single hand position data)
if re-capture the hand then
    clear the queue;
    fill the queue with new position untill full;
else
    get position;
    new position enqueue;
    dequeue;
end if
if hand position requested then
    return average;
end if
```

4 Game Platform

4.1 UI Config

Fig. 3 shows us that, from the start panel, we can enter two different scenes. In both situations we can control the character, or exit to the start.

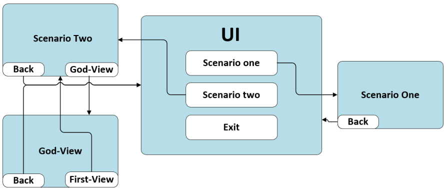


Fig. 3. Game platform structure

4.2 Scene Construction

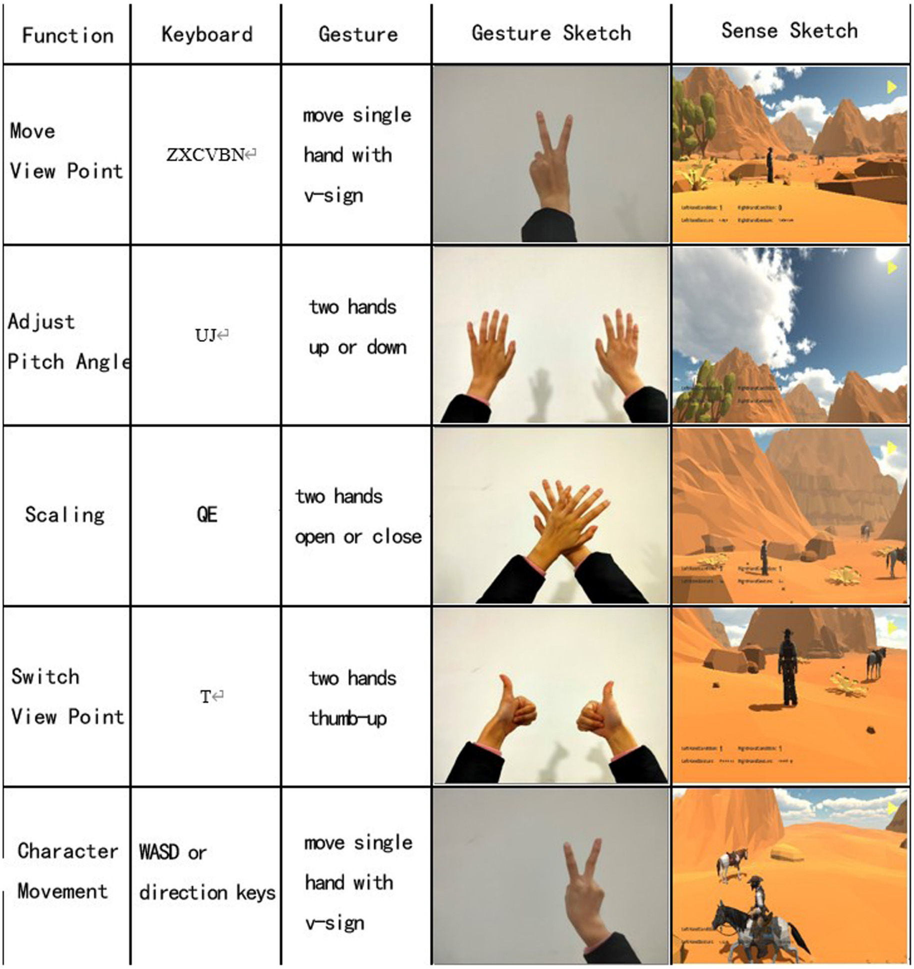
The .max files created by software 3Dmax are compatible with Unity. So, first, we constructed the scene assets in 3DMax, and then imported them into Unity. To get an acceptable viewing experience, we set up new parameters for main camera, directional lights, shadow, material shaders.

4.3 Hardware Config

Using RealSense SR300 and SDK 2016, we can get depth image schematic information and build a data controlling component.

4.4 Keyboard Event Mapping Design

The controlling system is based on Unity's embedded keyboard input events. To fulfill basic operations in the game scene, we mapped the gesture signal with keyboard events. See Fig. 4.



Function	Keyboard	Gesture	Gesture Sketch	Sense Sketch
Move View Point	ZXCVBN↵	move single hand with v-sign		
Adjust Pitch Angle	UJ↵	two hands up or down		
Scaling	QE	two hands open or close		
Switch View Point	T↵	two hands thumb-up		
Character Movement	WASD or direction keys	move single hand with v-sign		

Fig. 4. Controlling test example

5 Result Analysis

5.1 Effects of Shaking Optimization Algorithm

After applying the shaking optimization algorithm for gesture recognition, an overall smoother experience is obtained during interaction. Moving on to data visualization, as explained by Fig. 5, the red dots mean the original object position, and the blue dots mean optimized position. To obtain these results, we used left hand and kept still during the analysis.

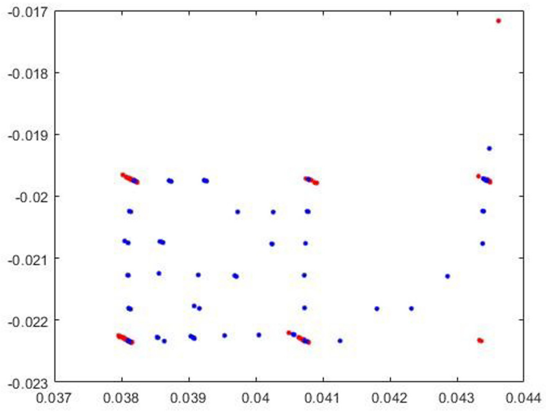


Fig. 5. Hand position differences between frames (Color figure online)

The conclusion that can be drawn from Fig. 6, is that original data had a clear regularity. This is due to maximum resolution of the depth camera. The optimized algorithm reduces the differences in hand positions between each frame, thus making the operation smoother. After designing the algorithm, we set several values for the size of position queue, including 2, 3, 4, and more than 5.

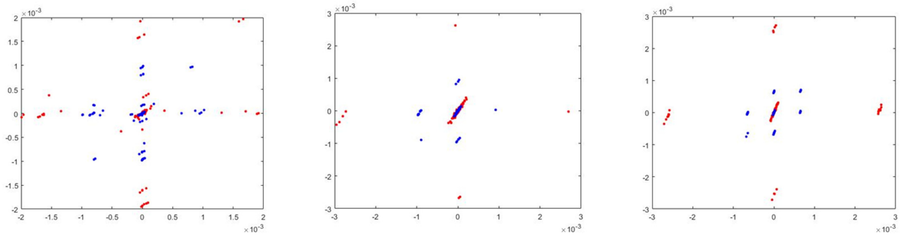


Fig. 6. Relative differences of original and optimized position data with queues sized 2 (left), 3 (mid), 4 (right)

The algorithm was also tested with the size bigger than 5, however, the optimizing effect was no longer discernible. Considering longer queue will cause more computing cost and larger delay, it was decided to finally set the size of queue as 5.

We used the same algorithm to dispose these data points, the result is proved to be good.

The tracking of the hand is much smoother. So, we can get better interaction experience under these conditions.

5.2 Limits of SR300 the Depth Camera

As the SR300 uses active near-infrared light as its structured light source, it runs quite well in dark environments.

However, when faced with bright environments such as outdoor scenes, the light will disturb the camera heavily. It can barely recognize the whole palm at a distance of 20 to 30 cm.

6 Conclusion

During this project, we experimented and analyzed the feasibility of building a gesture-based 3D scene exploring platform with existing camera hardware and game engine. With conclusive evidence we proved that this is a reasonable direction of man-machine interaction. As Internet of things and smart homes become more widespread, gesture controlling will have more numerous applications.

Acknowledgement. This work was supported in part by the National Key Research and Development Program of China under Grant 2018YFF0300903, in part by the National Natural Science Foundation of China under Grant 61872241 and Grant 61572316, and in part by the Science and Technology Commission of Shanghai Municipality under Grant 15490503200, Grant 18410750700, Grant 17411952600, and Grant 16DZ0501100.

References

1. Alisi, T., Bimbo, A.D., Valli, A.: Natural interfaces to enhance visitors' experiences. IEEE Multimed. **12**(3), 80–85 (2005)
2. Baraldi, S., Bimbo, A.D., Landucci, L., Valli, A.: wikiTable: finger driven interaction for collaborative knowledge-building workspaces. In: Proceedings of 2006 IEEE International Conference on Computer Vision and Pattern Recognition Workshop, p. 144 (2006)
3. Kaltenbrunner, M., Jordà, S., Geiger, G., Alonso, M.: The reactable: a collaborative musical instrument. In: Workshop on Tangible Interaction in Collaborative Environments (2006)
4. Kamel, A., Sheng, B., Yang, P., Li, P., Shen, R., Feng, D.D.: Deep convolutional neural networks for human action recognition using depth maps and postures. IEEE Trans. Syst. Man Cybern. Syst. **49**(9), 1806–1819 (2019)
5. Karambakhsh, A., Kamel, A., Sheng, B., Li, P., Yang, P., Feng, D.D.: Deep gesture interaction for augmented anatomy learning. Int. J. Inf. Manage. **45**, 328–336 (2019)
6. Ke, Q., Bennamoun, M., An, S., Sohel, F., Boussaid, F.: A new representation of skeleton sequences for 3D action recognition. In: CVPR, pp. 4570–4579 (2017)
7. Lu, P., Sheng, B., Luo, S., Jia, X., Wu, W.: Image-based non-photorealistic rendering for realtime virtual sculpting. Multim. Tools Appl. **74**(21), 9697–9714 (2015)
8. Meng, X., et al.: A video information driven football recommendation system. Comput. Electr. Eng. **85**, 106699 (2020)
9. Oreifej, O., Liu, Z.: HON4D: Histogram of oriented 4D normals for activity recognition from depth sequences. In: CVPR, pp. 716–723 (2013)

10. Siena, F.L., Byrom, B., Watts, P., Breedon, P.: Utilising the intel RealSense camera for measuring health outcomes in clinical research. J. Med. Syst. **42**(3), 1–10 (2018). https://doi.org/10.1007/s10916-018-0905-x
11. Sheng, B., Li, P., Zhang, Y., Mao, L., Chen, C.L.P.: GreenSea: visual soccer analysis using broad learning system. IEEE Trans. Cybern. 1–15 (2020). https://doi.org/10.1109/TCYB.2020.2988792. https://ieeexplore.ieee.org/document/9098099
12. Stefano, B., Alberto, D., Lea, L., Nicola, T.: Natural interaction. In: Encyclopedia of Database Systems. https://link.springer.com/referenceworkentry/10.1007
13. Vemulapalli, R., Arrate, F., Chellappa, R.: Human action recognition by representing 3D skeletons as points in a lie group. In: CVPR, pp. 588–595 (2014)
14. Vemulapalli, R., Chellappa, R.: Rolling rotations for recognizing human actions from 3D skeletal data. In: CVPR, pp. 4471–4479 (2016)
15. Wang, L., Huynh, D.Q.: Koniusz: a comparative review of recent Kinect-based action recognition algorithms. IEEE Trans. Image Process. **29**, 15–28 (2020)
16. Xia, L., Chen, C., Aggarwal, J. K.: View invariant human action recognition using histograms of 3D joints. In: CVPR, pp. 20–27 (2012)
17. Yang, X., Tian, Y.: Super normal vector for activity recognition using depth sequences. In: CVPR, pp. 804–811 (2014)
18. Zhang, P., et al.: Tracking soccer players using spatio-temporal context learning under multiple views. Multimedia Tools Appl. **77**(15), 18935–18955 (2018)
19. Zhu, Y., Fujimura, K.: Constrained optimization for human pose estimation from depth sequences. In: Asian Conference on Computer Vision [S.l.], pp. 408–418. IEEE Press (2007)
20. Shotton, J., et al.: Real-time human pose recognition in parts from single depth images. In: Computer Vision and Pattern Recognition. [S.l.], pp. 1297–1304. IEEE Press (2011)