

# THE KINECT SENSOR AS HUMAN-MACHINE-INTERFACE IN AUDIO-VISUAL ART PROJECTS

Matthias Kronlachner

Student, University of Music and Dramatic Arts  
Graz, Austria  
mail@matthiaskronlachner.com

Iohannes m zmölnig

Institute of Electronic Music and Acoustics  
University of Music and Dramatic Arts  
Graz, Austria  
zmoelnig@iem.at

## ABSTRACT

For several years now, the entertainment and gaming industry has been providing multifunctional and cheap human interface devices which can be used for artistic applications.

Since November 2010 a sensor called Kinect™ for Microsoft's Xbox 360 is available. This input device is used as color camera, microphone array, and provides - as an industry-first - a depth image camera at an affordable price. Soon after the release of the Xbox-only device, programmers from all over the world provided solutions to access the data from a multitude of operating systems running on ordinary computers.

This paper presents operating system independent ways to access and interpret Kinect data streams within Pure Data/Gem<sup>1</sup> and possible applications for interactive audio and visual art projects.

## KEYWORDS

pure data, kinect, depth sensor, human computer interface, motion capturing, interaction

## 1. INTRODUCTION

Based on a project work[4] multiple Pure Data/Gem externals<sup>2</sup> and application examples are introduced which allow the full access to the functionality of the Kinect sensor including video and audio streams.

So far no special colorspace is existent in Gem that describes the distance of every pixel in relation to the camera. Therefore, possible solutions are presented to interpret and work with the depth pixel data.

By integrating a framework for "natural interaction" (OpenNI), it is possible to do user identification or extract skeletal models from the depth image.

<sup>1</sup>Pure Data is a visual programming environment used for computer-music and interactive multimedia applications. Gem stands for *Graphics Environment for Multimedia* and extends Pure Data to do realtime OpenGL based visualizations.

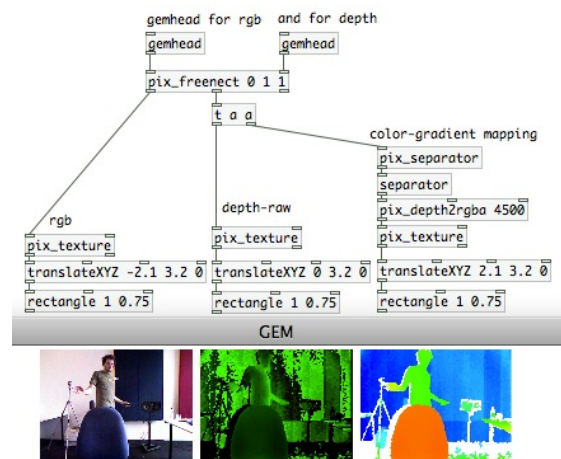
<sup>2</sup>Externals provide additional object classes to extend the functionality of Pure Data.

Other solutions exist to acquire skeleton data, for example OSkeleton<sup>3</sup>. However, the presented externals enable the access to video streams and tracking data simultaneously inside Pure Data without the need of additional computer programs. This allows maximum flexibility.

The documentation of the externals include usage examples like distance measurement, outline extraction, background subtraction, hand, user and skeleton tracking, head pose estimation as well as gathering audio streams from the 4 channel microphone array.

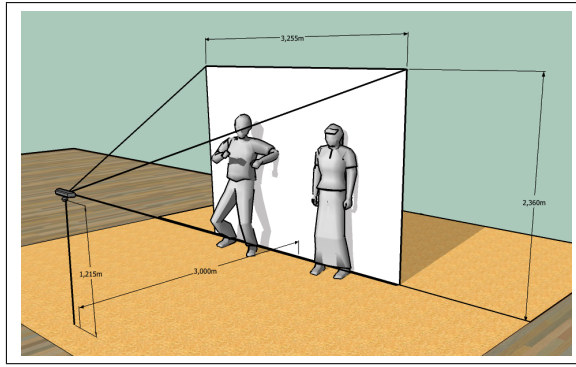
For showing the practical application in art projects, the piece *übersetzen - vertimas*[3] for dancer, sound and video projection as well as the usage for a concert with the IEM Computer Music Ensemble is presented.

The advantage of using Kinect for projects include independence from visible light due to operation in the infrared spectrum and no need for calibration to gather movement data of people.



**Figure 1.** Pure Data/Gem screenshot showing RGB stream, raw depth stream and color gradient mapping of the depth stream

<sup>3</sup>OSkeleton is a standalone application to gather skeleton data from OpenNI/NITE framework and sends it over Open Sound Control (OSC) to a host.



**Figure 2.** Depth sensor field of view 58° H, 45° V, 70° D

## 2. KINECT SPECIFICATIONS

The Kinect sensor includes an RGB camera with a standard resolution of 640x480 at 30 Hz framerate (max. 1280 x1024 @ 15 Hz), a depth image sensor and a four channel microphone array running with 16 bit resolution and 16 kHz sampling rate. It's head can be tilted  $\pm 27^\circ$ , a three axis accelerometer is measuring the orientation and a three color LED can be used for visual feedback.

The depth sensor consists of an infrared laser projecting a specific dot pattern onto it's field of view. An infrared camera records these patterns on the objects and an on-board DSP computes the distance by correlating the live image with stored reference patterns. Using multiple Kinect devices onto the same scene can cause unrecognized regions in the depth image by overlapping patterns. A possible solution by adding independent motion to each of the sensors is proposed in [Mai2012][5].

The output of the depth sensor is a 640x480 pixel video stream, each pixel holding 11 bit of depth information.

## 3. ACCESSING DATA STREAMS

Several software libraries exist to allow programmers interface with the Kinect sensor.

### 3.1. libfreenect

The OpenKinect community released the multi-platform open source library libfreenect, allowing access to all data streams of the Kinect. No higher level functions are included but there is a separate open source project focusing on skeletal tracking (Skeltrack[6]).

### 3.2. OpenNI/NITE

The Israeli company Primesense which developed the technology for Kinect, founded the non-profit organization OpenNI to support the interoperability of Natural Interaction devices. OpenNI is a cross-platform open source framework allowing applications to interface with different sensor devices and accessing higher level functions (middleware) like skeleton tracking. The built-in function for recording and playing back data streams of an attached

sensor device is handy for rehearsal and development situations.

NITE is the closed source middleware from Prime-sense and provides gathering of the position of numerous standing people, tracking of a detailed skeleton of two people as well as performing hand tracking.

### 3.3. Microsoft Kinect SDK

In June 2011 Microsoft released their Kinect Software Development Kit (SDK). This Windows only SDK allows position estimation of up to six standing people and extracting the detailed skeleton of two people. It features higher level functions for the microphone array like sound source localization, beam forming and speech recognition.

## 4. PD EXTERNALS

### 4.1. Representation of depth data

So far no Gem colorspace exists which describes the distance of a pixel in relation to the camera. Therefore a solution is proposed using RGBA or YUV colorspace for representing 16 bit depth data. For RGBA output the 16 bit depth data is divided into the upper eight most significant bits and the lower eight significant bits. These eight bit values are stored in the red (R) and green (G) channel. The blue channel (B) is used for additional information about the pixel. For example, if a user is present in that specific pixel, the specific user-id is set. The alpha channel (A) is set to 255. YUV colorspace uses 4 bytes per 2 pixels and therefore can store the 16 bit per pixel depth information, but additional values like user-ids can not be included.

R	G	B	A
3/8 msb	8 lsb	0 or userid (OpenNI)	255

**Table 1.** RGBA output of depth data

YUV422 (2 bytes per pixel)
11 bit/16bit depth values

**Table 2.** YUV output of depth data

Depending on the numerical representation of color values, depth information can be obtained with the following formulas<sup>4</sup>.

$$distance = R_{int} * 2^8 + G_{int} \quad (1)$$

$$distance = R_{float} * 2^{16} + G_{float} * 2^8 \quad (2)$$

<sup>4</sup>Gem internally handles color values as 8 bit integers (0-255), but on user level normalized floats (0.0-1.0) are used.

For development and visualization purposes it is handy to map distance values onto a color gradient using `pix-depth2rgba`.



**Figure 3.** Gradient for displaying depth data - near to far

## 4.2. `pix_freenect`

The external `pix_freenect`[2] is based on `libfreenect` and offers access to RGB and depth streams (Fig. 1). It allows gathering of accelerometer data, controlling the tilt of the head and changing the LED color. Multiple devices can be accessed by their unique serial number. Due to the limited bandwidth of USB 2.0 a maximum number of two Kinects can be accessed by one USB controller.

The message `depth_mode` gives the possibility to choose between output of a depth value in [mm], an RGB-aligned [mm] output and the raw 11 bit data from the depth sensor.

## 4.3. `pix_openni`

Based on OpenNI and NITE middleware by Primesense, the external `pix_openni`[2] features some higher level functionality. Besides gathering the RGB and depth streams it is possible to do hand tracking, user tracking and skeleton tracking (Fig. 4). Currently it is not possible to get accelerometer data, control the tilt and the color of the LED as well as receiving the audio streams. Therefore the `libfreenect` based externals `freenect`[2] and the currently Linux only `freenect_audio`[2] have been developed to be used simultaneously with `pix_openni` and provide the missing features.

# 5. APPLICATION EXAMPLES

## 5.1. General application examples

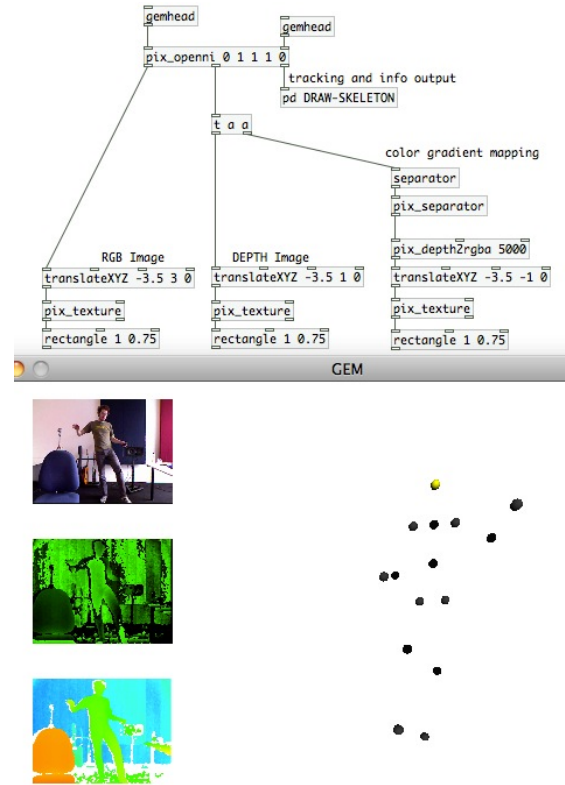
The following application examples for Pure Data/Gem are available on Github[2].

### 5.1.1. Distance measurement

By using `pix_data` the color of a specific pixel can be extracted. Using the formulas described in 4.1 it is possible to convert RGBA color data back into a distance value.

### 5.1.2. Background subtraction

By defining three dimensional high and low level thresholds, a cuboid of the depth map can be extracted. Computation can either be done with the help of `pix_threshold_depth`[2] (calculation done on CPU) or with OpenGL shaders (calculation done on GPU). Overlaying the obtained stencil with the aligned RGB image results in masking unwanted regions.



**Figure 4.** Skeleton tracking with `pix_openni`

### 5.1.3. User tracking

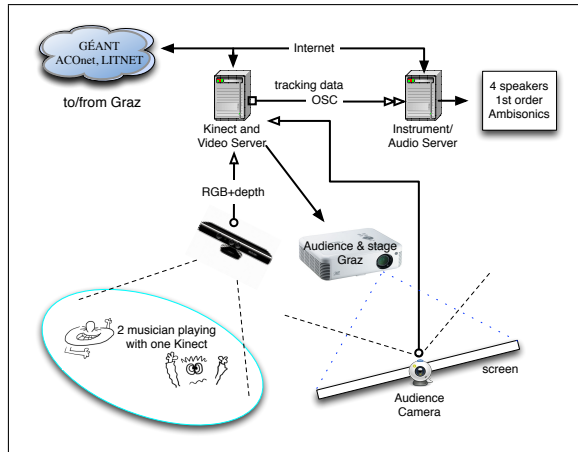
Two different approaches for tracking a user are included in the examples. The general approach would be to use the processed depth map of section 5.1.2 and get the position and size of the remaining objects (blobs) in the depth map by using `pix_multiblob`.

`pix_openni` gives access to higher level scene analysis algorithms provided by the NITE middleware. It can output the center of mass of numerous standing people. As described in section 4.1 the B channel of the RGBA depth-map is used to store a user-id depending on the presence of a user in the specific pixel. User-ids are assigned sequentially starting at one. After ten seconds of absence the user-id is freed and can be reused if a new user appears in the field of view. Filtering the depth map according to the B channel is an easy way to extract the outline of all people in the field of view.

### 5.1.4. Skeleton/hand tracking

Once the NITE algorithm detected a user it can start tracking a detailed skeleton for two users without the need of a calibration pose. The tracking data consists of three-dimensional coordinates for each of the 15 joints (Fig. 4). NITE also supports the tracking of multiple hands. After making a waving gesture it starts to output three-dimensional coordinates of the hand.

Due to the frame rate of the depth sensor the maximum output rate of the tracking data is limited to 30 Hz.



**Figure 5.** ICE network concert - stage setup Vilnius

All tracking data is represented in [mm] real-world coordinates with their origin being at the infrared camera.

#### 5.1.5. Head pose estimation

Based on a paper and software by Gabriele Fanelli[1] the external `pix_head_pose_estimation`[2] has been developed which takes the depth map of the Kinect as input and estimates the Euler angles and position of multiple heads detected in the depth map. The estimator works with a reference database and covers a range of about  $\pm 75^\circ$  yaw and  $\pm 60^\circ$  pitch.

### 5.2. ICE - IEM Computermusic Ensemble

ICE<sup>5</sup> is a group of electronic musicians, each playing with a notebook and individual controllers. The target is to play contemporary music, adapted or written for computermusic ensembles.

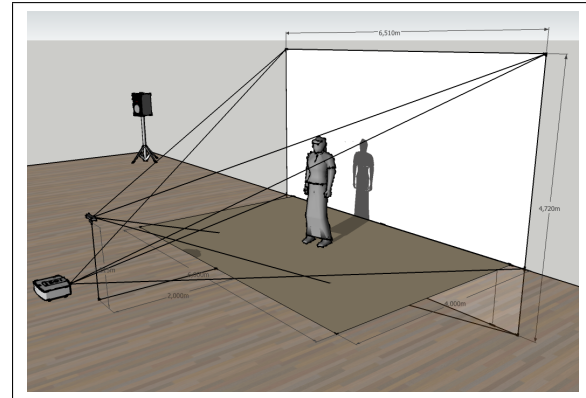
In March 2012 a network concert between Graz and Vilnius took place. One Kinect sensor was used in Vilnius to track the skeleton of two musician. The tracking data allowed each musician to play his virtual instrument without handheld controllers. Additionally the Kinect video stream showing the stage in Vilnius was sent to Graz and projected on a canvas for the remote audience.

### 5.3. vertimas - übersetzen

The piece *übersetzen - vertimas*[3] for dancer, sound and projection developed by the author features the Kinect sensor to translate body movements on stage into sound and turns the dancers body into a hyperinstrument. Additionally, the depth video stream is used to gather the outline of the dancer and project back onto her body in realtime (Fig. 6).

Therefore an data-flow filtering and analysis library has been developed to enable quickly adjustable methods to translate tracking data into control data for sound or visual content.

<sup>5</sup>IEM - Institute of Electronic Music and Acoustics, Graz ICE: <http://www.iaem.at/projekte/ice>



**Figure 6.** stage setup *vertimas - übersetzen*

## 6. OUTLOOK

Using Kinect allows skeleton tracking without the need of body mounted sensors or reflectors. This makes the usually technoid flavor of an interactive performance invisible and creates some more mysteries about the human-computer-interaction used. The different data streams of the sensor give many possibilities to create a bridge between art installations and their visitors just by using a single USB device.

The sensor device can be bought in almost every electronic shop around the world. Therefore easy replacement during tours is guaranteed.

The cons of using Kinect include the limited range and resolution, the possible interference of other infrared light sources and the momentary dependance on non-open source software for higher level functionality.

## 7. REFERENCES

- [1] G. Fanelli, T. Weise, J. Gall, and L. V. Gool, "Real time head pose estimation from consumer depth cameras," in *33rd Annual Symposium of the German Association for Pattern Recognition (DAGM'11)*, September 2011.
- [2] M. Kronlachner. (2012, 08) Source code repository. [Online]. Available: <http://github.com/kronihias>
- [3] M. Kronlachner. (2012, 04) *übersetzen - vertimas* - piece for dancer, sound and projection - trailer. Vilnius, Lithuania. [Online]. Available: <http://vimeo.com/40919205>
- [4] M. Kronlachner, "The kinect distance sensor as human-machine-interface in audio-visual art projects," Institute of Electronic Music and Acoustics, Graz, Tech. Rep., 2012.
- [5] A. Maimone and H. Fuchs, "Reducing interference between multiple structured light depth sensors using motion," *IEEE Virtual Reality 2012*, March 4-8, 2012.
- [6] J. Rocha. (2012, 07) Skeltrack. [Online]. Available: <http://github.com/joaquimrocha/Skeltrack>