LiveScan3D manual (a work in progress)

LiveScan3D is a software designed for real time 3D reconstruction using multiple Kinect v2 depth sensors simultaneously at real time speed. The produced 3D reconstruction is in the form of a coloured point cloud, with points from all of the Kinects placed in the same coordinate system. Possible use scenarios of the system include:

- capturing an object's 3D structure from multiple viewpoints simultaneously - Figure 1,
- capturing "panoramic" 3D structure of a scene – Figure 2,
- streaming the reconstructed point cloud to a remote location,
- increasing the density of a point cloud captured by a single sensor, by having multiple sensors capture the same scene.

At the moment connecting multiple Kinect v2 devices to a single computer is difficult and only possible under Linux.

Because of those limitations, in our system each Kinect v2 sensor is connected to a separate computer. Each of those computers is connected to a server which governs all of the sensors. The server allows the user to perform calibration, filtering, synchronized frame capture, and to visualize the acquired point cloud live. Sequence of clouds, captured in real-time, can be stored in .ply files

Figure 1: A setup for capturing an object from multiple viewpoints.

Figure 2: A setup for capturing a panoramic 3D reconstruction.

Step by step configuration (Tutorial 1)

In this tutorial we describe step by step how to configure and run LiveScan3D with Kinect v2 sensors in a scenario where all Kinects are able to see a single calibration marker. In order to start you will need to:

- have at least one Kinect v2 sensor,
- download LiveScan3D from http://ztv.ire.pw.edu.pl/mkowalski/ where it is available as a compiled binary or compile it yourself from source code available on GitHub http://github.com/MarekKowalski/LiveScan3D/
- download and install the Kinect for Windows SDK 2.0 on each machine you intend to use as a client,

- download and install the Visual C++ Redistributable for Visual Studio 2013,
- have all of the computers you will use in the same network,
- print the calibration pattern “calibration0.jpg” on a piece of paper (A4 size should be enough).

Once you have all of the preliminary steps completed, run the LiveScanClient application on each of your client computers. If everything is working fine, you should see the RGB camera stream of your Kinect inside the application window.

Next, choose a computer that will act as a server (this computer may also be a client at the same time), we recommend that it is the most powerful of the available machines. Run the LiveScanServer on the chosen machine and click “Start server” to begin listening for client connections. Connect each of the clients to the server. If the client is running on the same machine as the server, there is no need for inputting an IP address (it defaults to 127.0.0.1).

At this point your server window should show some clients connected. You can press “show live” to see the output from the sensors. Here you will notice two things, first of all the point clouds from different devices are not aligned, second of all the frame rate might be low. The first problem arises, because the clients are not “calibrated”. By calibration here I mean “knowing the location of the Kinect sensor in the scene”. As for the low FPS in the live view window, please read the section of the manual about this window.

In order to calibrate the clients you need the printed calibration pattern. Once printed, attach it to something rigid and place it in a position where it is visible to all sensors. Note that the calibration pattern (marker) must be visible to the sensor’s depth and color stream, you can check if it is by pressing “show depth” in the client window.

Now there is only one last thing you need to do, which is to make sure that the server knows which marker you want to use. In order to make sure that is the case go to settings and make sure that under “calibration markers” you have a maker with id 0. Now all you have to do is press calibrate and the data from you Kinects should align. You can check if that is the case in the live view window.

LiveScanServer

The LiveScanServer application governs the clients and allows for calibration, filtering, sequence recording, live preview of the reconstruction etc. Below you will find a separate section for each window of LiveScanServer with explanations of each of the program's functionalities.

Main window

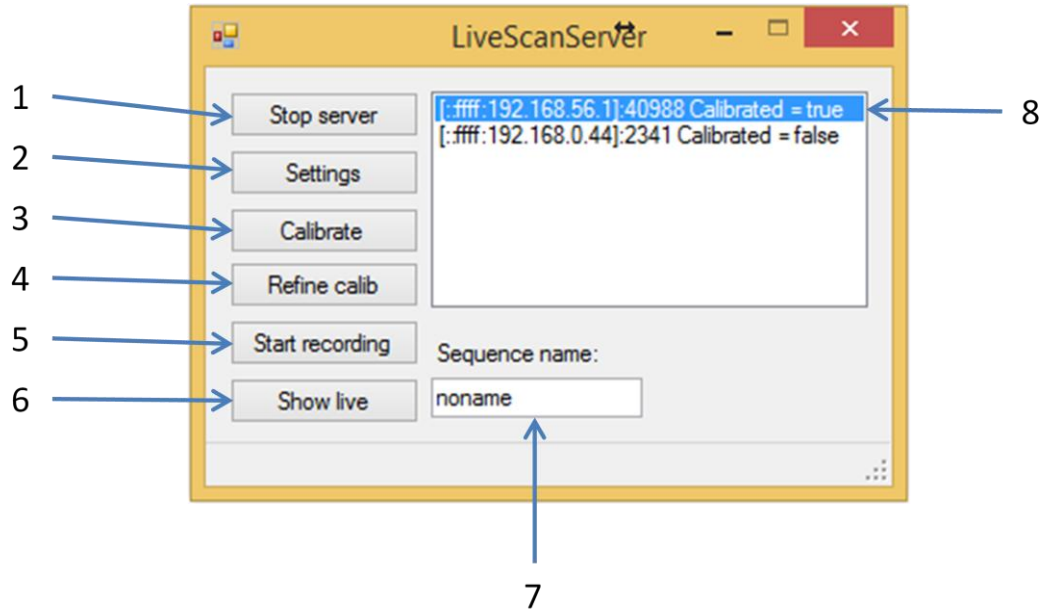

Figure 3: The main window of the server application. It shows two clients connected, one of which is calibrated.

1. Starts and stops the server – the server needs to be started before any work can be done. If there are any errors when trying to start, make sure that this is the only instance of LiveScanServer running and that there are no apps using port 48001.
2. Opens the settings window – more details below.
3. Performs calibration using markers – each client performs the calibration separately and once it succeeds that client show "Calibrated = true". In order to perform calibration the clients need to know the locations of the markers in the scene, those locations are set in the settings menu. If a client sees many markers, the largest one is always used.

   In some scenes where not all Kinects are able to see the same marker, you will need to use more than one. There are six different marker patterns included in the project files. Each of them corresponds to a unique id of 0 to 5. In order to use a marker with a given id it needs to be added to the set of known markers in the settings window.

   If one of the clients does not calibrate, there can be several reasons:
   - The marker is not getting detected – this is fairly common if there are bad light conditions. Try and move it around, or switch the light on in the room to see if it helps.
   - The marker's pose is not known to the client – this usually manifests by a green border around the marker (which indicates that it is correctly detected), but no calibration. To fix this, simply add the marker with the proper id in the settings menu.

4. Refines the current calibration – if all of the sensors are already calibrated and there still some misalignment between the point clouds, this functionality may help. It uses Iterative Closest Points (ICP) to refine the alignment between the point clouds. This function only works well if there is a fair deal of common surfaces between the point clouds from different sensors. You can use it multiple times for further improvement.
5. Performs recording – once this button is pressed the clients begin to capture and locally store frames in a synchronous manner. Once you press this button again, the recording will stop and the clients will begin uploading the frames to the server. The server will save them as PLY files in a directory specified by "sequence name". Pressing the button again before all frames are saved will stop saving and allow the recording of a new sequence.
6. Opens the live view window – more details below.
7. Sequence name – this specifies the directory to which the PLY files will be saved if you perform recording. The whole directory path will be "./out/<sequence name>".
8. List of all of the connected clients – it shows the client IP addresses along with the information on whether they are properly calibrated or not.

Setting window

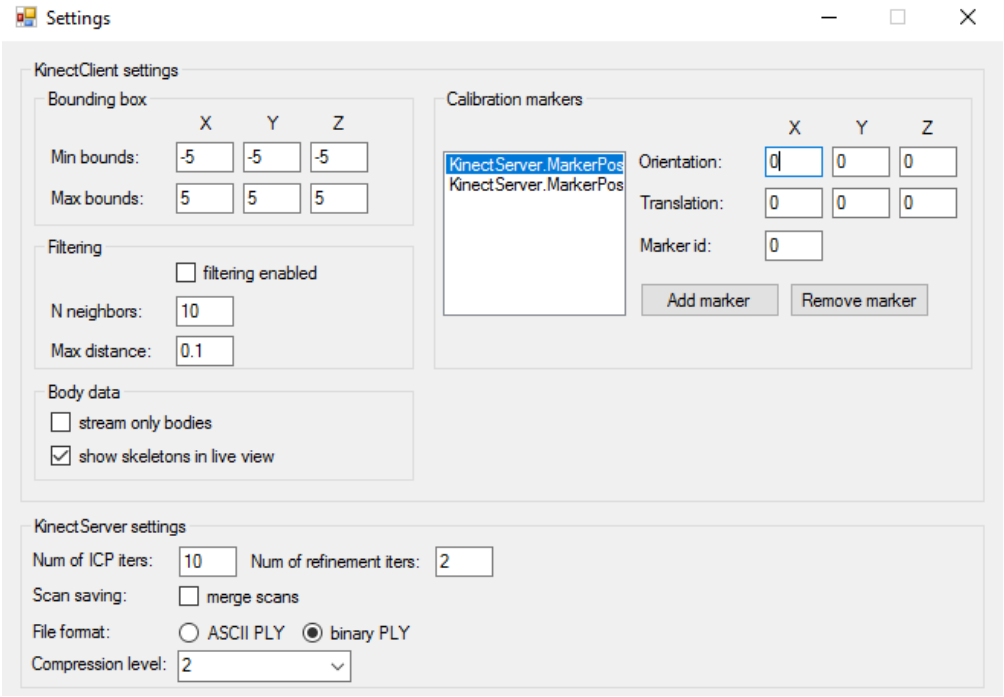

Figure 4: The settings window of the server application. Two markers are defined.

The most important part of the settings window is the **calibration markers** panel. Here you can define the positions of the markers in your scene, which will allow you to define the world coordinate system. The marker orientation and translation describe the marker's pose in the world coordinate system (orientation is in degrees, translation in meters). The marker id allows the program to identify individual markers, the marker's id is the number in its file name in /docs/calibration markers/. Using multiple markers is necessary in situations where not all off the sensors can see a single individual marker.

The **bounding box** panel allows for removing parts of the scene which are not necessary. The points outside of the bounding box are removed in the client application and never reach the server. This is

important as it reduces the bandwidth between the clients and the server and thus increases the frame rate in the live view window. All the coordinates are in meters.

The **filtering** can be applied to remove noise and flying pixels. The filtering will remove any point which has its Nth neighbour further than max distance. In order to filter more points you can either decrease the max distance or increase the number of neighbours. It is important to remember that enabling filtering puts a lot of strain on the client computers and might decrease frame rate.

The **body data** panel controls how body and skeleton data is handled. If stream only bodies is chosen the clients will only send the points that are associated with detected bodies. This is very useful for noise removal and it increases the frame rate as less data is transmitted between the clients and the server.

The **KinectServer** panel controls some of the settings of the server:

- Num if ICP iters, num of refinement iters both control calibration refinement, for details refer to [1].
- Scan merging controls whether the point clouds from separate clients are saved into a single, merged file or to separate files (this occurs after recording).
- File format controls the type of PLY that will be saved. The binary PLY format is much more compact and can be opened using the LiveScanPlayer.
- Compression level controls how much compression is applied to the data sent from the clients to the server. For details on the meaning of the levels refer to ZSTD's documentation.

Live view window

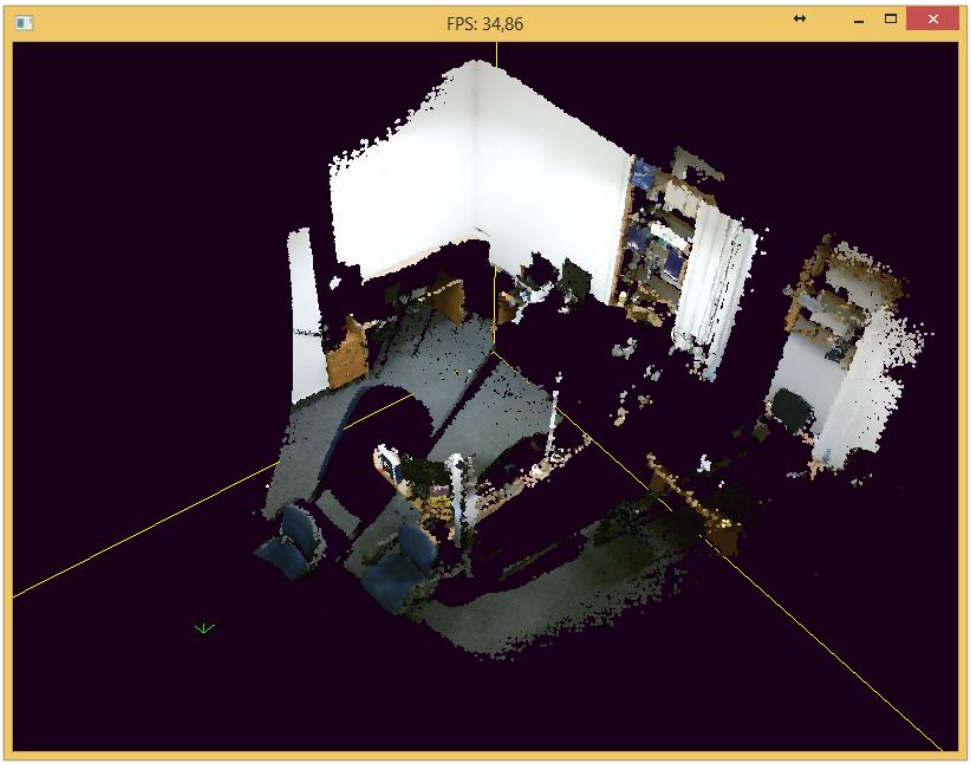

Figure 5: The live view window, the number of frames per second (FPS) is visible in the window's title.

This window shows the reconstructed point cloud from all of the sensors. Apart from the points themselves you will also find the marker positions in the scene as red lines and sensor positions as green lines. The frame rate here depends greatly on the speed of you network and on the number of devices you are using. In the future (hopefully near future), we plan to work on reducing the size of the point clouds, which should increase the FPS.

Remember than even when there are only a few frames per second in this window, the frame rate should be high if you record the frames (using the "Start recording" button in LiveScanServer).

There is a number of ways to move around the reconstructed point cloud in this window:

- Left mouse button – lets you rotate the point cloud,
- Right mouse button – lets you move the camera,
- Mouse wheel – zoom in/zoom out,
- W, S, A, D keys – let you move the camera around like in an FPS game,

There are also additional functionalities under the following keys:

- +, - keys – change the size of the points.
- F key – toggle full screen mode
- M key – toggle marker and bounding box visibility
- P, O – increase, decrease brightness of the point cloud

LiveScanPlayer

This application allows for viewing and transmitting the pre-recorded sequences, which were saved by the clients (.bin format) or the server (.ply format). It also allows for saving data from the .bin files in the .ply format.

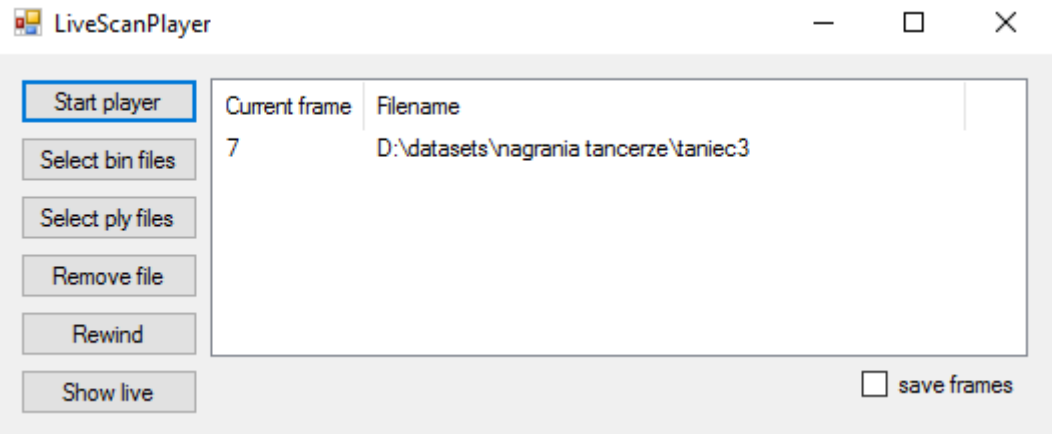

Figure 6: The main window of the LiveScanPlayer.

- **Start player** – starts or stops the player and the Transfer Server. The Transfer Server allows for streaming the played data to a remote device, for example the Hololens using our Unity application. If **save frames** is selected during playback, the played frames will be saved as binary PLY in the ./outPlayer/ directory.
- **Select bin files** – allows for opening the bin files created by the clients during recording. Each bin file contains an entire sequence of frames.
- **Select ply files** – allows for open binary PLY files saved using the server. It is not guaranteed that it will allow opening PLY files saved by other software. Each PLY file contains a single frame so you need to select multiple files to have a sequence.
- **Remove file** – removes the selected sequence from the list. Any file can be selected by clicking on the current frame counter.
- **Rewind** – sets the current frame of all sequences to 0.
- **Show live** – shows the live view window, which works identically to the same window in the server.
- **Changing the value of the current frame** – makes the chosen sequence skip to that frame.

Bibliography

[1] Kowalski, M.; Naruniec, J.; Daniluk, M.: "LiveScan3D: A Fast and Inexpensive 3D Data Acquisition System for Multiple Kinect v2 Sensors". in 3D Vision (3DV), 2015 International Conference on, Lyon, France, 2015