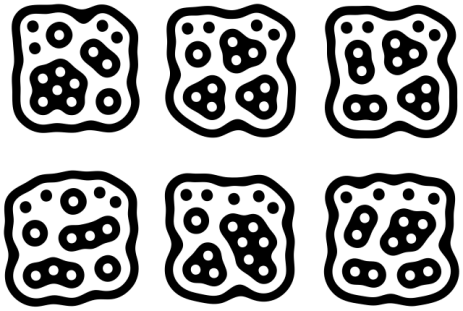
Uniducial User Guide

Uniducial is a library to simplify the development of Unity real-time applications for table-based tangible user interfaces. It has been developed during a project at HTW Berlin 2009/10 when we explored various possibilities to physically interact with computer games.



One way to physically interact with games are fiducial markers. Fiducials are specially designed markers that can be attached to a physical object to track its position and rotation. Although they were originally invented to be used on table-based tangible user interfaces (TUI), we investigated how they could be used to track movement and actions of a player.
Although this approach turned out to be non intuitive and too error prone for most games, we successfully used it to steer a car in a racing game by attaching one fiducial marker to a real steering wheel.

During the development of several game prototypes we noticed that the development for our Unity based prototypes could be simplified by creating a library that takes care of various repetitious tasks. Uniducial provides

- Connecting to ReactiVision
- Position mapping
- Rotation mapping
- Marker management
- Access to other marker attributes such as velocity, acceleration, rotational velocity etc.

Uniducial is released under the MIT license which means that distribution and modification for closed or open source projects is very easy.
You can find all updates to Uniducial, example applications and the code repository at http://code.google.com/p/uniducial/

For more information on fiducial markers and how to build a table-based tangible user interfaces table see http://reactivision.sourceforge.net/

Requirements

To use Uniducial the following soft-/hardware is needed:

- a camera connected to your computer
- Printed fiducial marker attached to a physical object
- reacTIVision vision engine 1.4
  http://reactivision.sourceforge.net/
- Unity Free 2.6 or Unity Pro 2.6
  http://unity3d.com/
- a TUI table

Although we successfully used fiducials with no table at all and just one ordinary webcam, we do not recommend this method . It is too unreliable as successful marker recognition depends too much on your lighting conditions and on the distance from marker to camera. We had to use fairly huge markers in our prototypes to provide a solid recognition for interaction.
So bottom line is: Use fiducial for what they were invented for: object tracking on a table based tangible user interface.

Camera

When using a camera for a TUI application a few things need to be considered:

- The camera should have at least a resolution of 640 * 480 pixels and a USB2 connection to the PC. Higher resolutions might require a FireWire connection.
- The vendor's camera software should provide the possibility to manually set the camera's shutter speed and gain in order to get good tracking results. Any automatic adjustments like brightness or contrast control should be switched off in the camera software.
- For sharp images the lens needs to be focused.
- Use high shutter speed of 1/250s or 1/500s to avoid motion blur. ReactiVision fails to detect the fiducial markers correctly if the images are too blurry.
- Increase the camera gain or increase marker illumination.
  As higher shutter speeds result in less light arriving at the camera's light sensor, either the gain needs to be increased or brighter light sources are needed. Higher gain means that the camera is more sensitive to light, but it will also increase noise in the resulting image.
- Be aware that consumer cameras often use low quality wide angle lenses which tend to be blurrier at the edges. Therefore markers that are close to the edges of the camera's field of view may not be detected correctly.

reacTIVision

reacTIVision is a standalone application for tracking fingers and fiducial markers. It processes the video stream images sent by the camera and tries to recognize the markers.
It acts as a server, that sends messages via the TUIO protocol including information on each marker's ID, position and orientation. Any client application can connect to reacTIVision to further process the data.

Uniducial requires that reacTIVision is running and using the default UDP Port 3333 on localhost (172.0.0.1).

Uniducial

Uniducial consists of the following components:

- a C# library which handles the TUIO communication with reacTIVision and keeps track of fiducials (uses the reacTIVision C# TUIO client)
- the FiducialController component
- User Interface scripts to nicely integrate Uniducial in the Unity UI

Usage

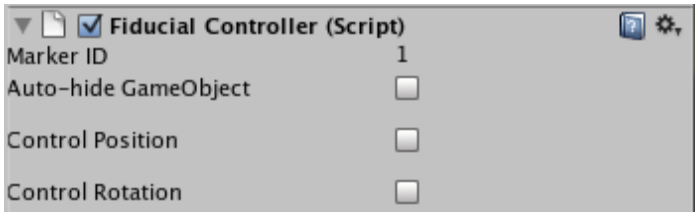
For each Unity project you need to copy the scripts as well as the library into the project's folder. The easiest way is to download the example project and copy the necessary folders to your project. Copy and paste:

- `/Uniducial/assets/Uniducial` to `/yourProject/assets` (library and component)
- `/Uniducial/assets/Editor to /yourProject/assets` (User Interface scripts)

Please notice that, while the library and component (UniducialLibrary.dll and FiducialController.cs) can reside anywhere in the assets folder, the GUI scripts need to be in /assets/Editor.

To use Uniducial in Unity:

1. Create a *GameObject* (Either empty or a preset) from the GameObject Menu
2. With the *GameObject* selected, add the *FiducialController* script from the Component->Input Menu
3. Adjust the *FiducialController* settings in the Inspector view



Marker ID

First thing you should do is to set the marker's ID, that is supposed to control the *GameObject.* Uniducial requires the use of the default marker set that comes with reacTIVision, and includes 215 markers.

Auto-Hide GameObject

Enabling this option hides the *GameObject* when the marker is no longer visible. For example when you move the fiducial object off the table, the *GameObject* disappears.

Control Position

Checking this option enables position mapping. Uniducial can map the marker position to any 3D *GameObject*, 2D *GUI Texture* and *GUI Text*.



Depending on the way you set up your TUI table and the camera, the images of the video stream might be flipped. You can use the **Invert-X** / **Invert-Y** option to correctly reflect the position of the fiducial marker.

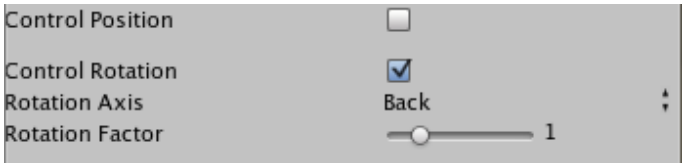
All marker driven *GameObjects* are translated on a plane in front of the *MainCamera*. No matter how the camera is oriented, they will always move in that plane. This means that your GameObjects are always at a constant distance to the camera.

If the main camera is set to perspective, the translation plane can be offset with the **Camera offset** slider. Altering this value will change the size of your *GameObjects*.

If you are using Unity to create a 2D scene you most likely set your main camera to orthographic projection. As there is no perspective distortion with orthographic cameras, the distance from a *GameObject* to the camera does not influence its size.

Controlling Rotation

To map the rotation from a marker to a *GameObject* check the **Control Rotation** option. Once enabled you can define the objects local **Rotation Axis**. If you need your objects to rotate slower or faster, you can alter the speed by changing the **Rotation Factor**.



Accessing marker attributes

For most TUI applications you will want to go beyond simple position and rotation mapping. For example you might want to apply forces to rigid bodies based on the markers velocity, change the color of a material or a play sound.
There are a lot of possibilities in Unity and there is no way that a single component could provide all the functionality you need to create what you imagine. To do the really cool stuff, you need to program.

Once the *FiducialController* is attached to a GameObject, you can write your own scripts and attach them to extend functionality and access other markers attributes, such as:

- Speed
- Acceleration
- Direction
- Rotational speed
- Rotational acceleration
- Angle (in radians or degrees)

Happy coding! ☺