d-track Documentation

Release 0.0

Ricardo Ribeiro

May 25, 2018

Getting started

D-Track is a suite of software used to track an object in 3D based on the 3D schematics of a scene. These set of software was used to calculate and analyse the 3D position of a dolphin in a pool in Zoomarine @ Portugal.

D-Track applications

d-track-singlecam	Extracts 2d information, used for the tracking, from a recording captured by a single camera.
d-track-smoothpath	Combine the 2 cameras 2d information to find the 3d path.
d-track-render	Render the resulting 3d path into a 3d scene.

Other software used in the project

Python Video Annotator	Software used to correct values from the 2d tracking.
Python 3D Engine	Python library for 3d physics simulation.
Python 3D Scene Editor	Software to create 3d scenes to be used with the py3dengine library.
3D-tracking-analyser	Software to visualize and analyse the results of the 3d tracking.

CHAPTER 1

Developers & collaborations

Ricardo Ribeiro	**from the Champalimaud Scientific Software Platform** ricardo.ribeiro@research.fchampalimaud.org
Patrícia Rachinas Lopes	**from the MARE-ISPA** plopes@ispa.pt
Dolphins	Zoomarine @ Portugal


CHAPTER 2

Code on Github

2.1 Install

Note: Currently only the environment configurations for Ubuntu 17 and MacOS X are available.

- Download and install Anaconda or Miniconda
- Download the environment configuration file for Ubuntu 17 or for the MacOSx.

```
#command to use on Ubuntu 17.10
wget --no-check-certificate https://raw.githubusercontent.com/UmSenhorQualquer/d-
↪track/master/environment-ubuntu17.yml

#or

#command to use on MacOS X
conda install wget
wget --no-check-certificate https://raw.githubusercontent.com/UmSenhorQualquer/d-
↪track/master/environment-macosx.yml
```

- Open the terminal and execute the next command to install the python environment.

```
#environment to use on Ubuntu 17.10
conda env create -f environment-ubuntu17.yml

#or

#environment to use on MacOS X
conda env create -f environment-macosx.yml
```

- Activate the environment.

```
source activate d-track-environment
```

- Clone the d-track repository.

```
git clone --recursive https://github.com/UmSenhorQualquer/d-track.git
```

- Change the current directory to the d-track folder.

```
cd d-track
```

- Run the installation script.

```
python install.py
```

2.2 Run D-Track

Note: Please make sure you have installed the environment before trying to execute the application.

- Open the terminal and execute the next command in the terminal to activate the environment.

```
source activate d-track-environment
```

2.2.1 Windows mode

- Execute the next command to open the Tracking application.

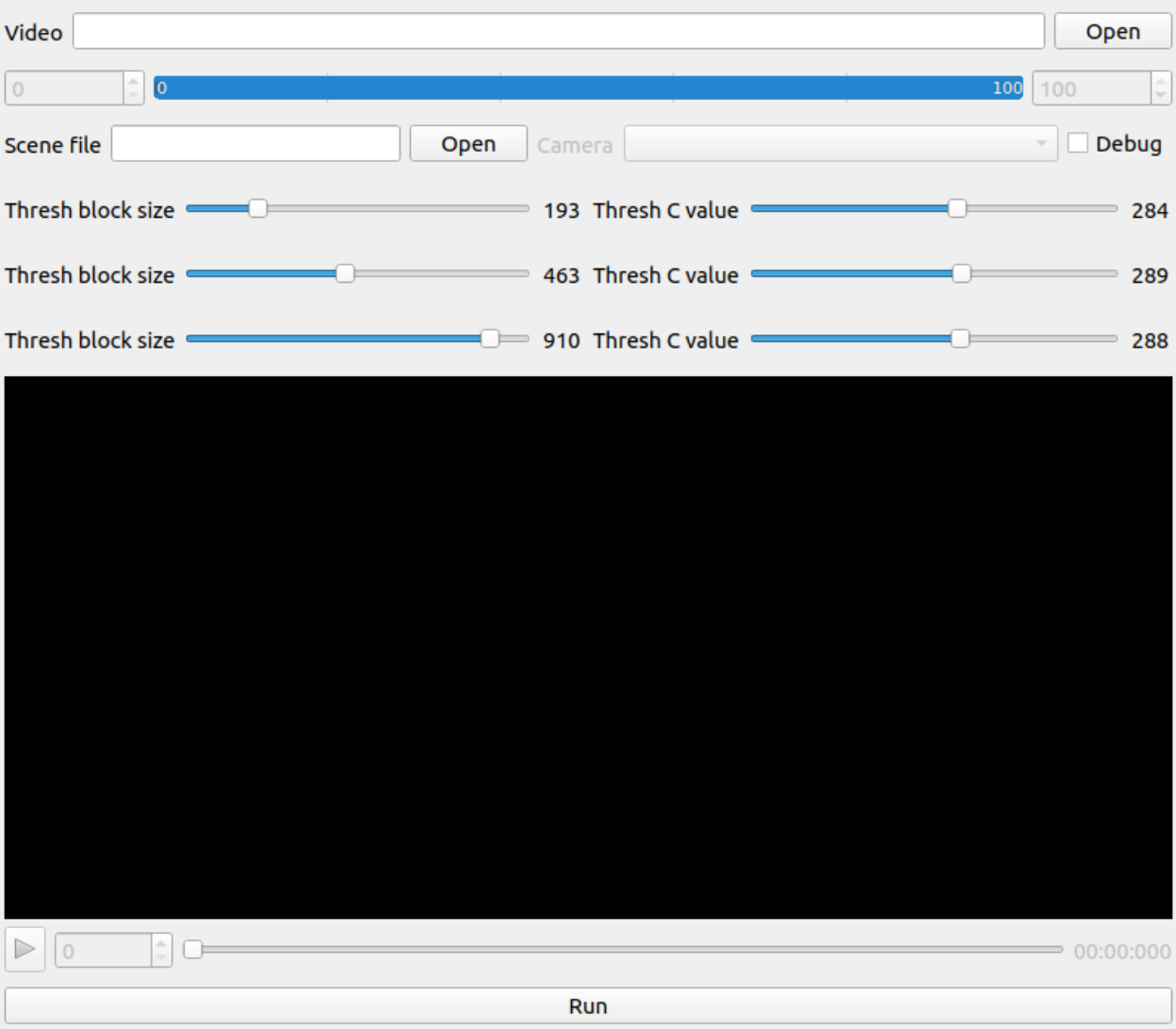
```
d-track-singlecam
```

- Or execute the next command to open the Smooth Path application.

```
d-track-smoothpath
```

- Execute the next command to open the Render application.

```
d-track-render
```



2.2.2 Batch/Terminal mode

Executing the software in batch mode is useful to analyse the videos on a computational cluster or to configure analysis to run one after each other using a shellscript.

To activate the batch mode, thanks to the PyForms framework, you just need to add the next 2 parameters to the applications calls.

```
d-track-singlecam terminal_mode --exec execute
```

or

```
d-track-smoothpath terminal_mode --exec execute
```

or

```
d-track-render terminal_mode --exec execute
```

More parameters

Call the command **help** to now which parameters you can use more.

```
d-track-smoothpath terminal_mode --help
```

```
usage: d-track-singlecam [-h] [--_sceneFile _SCENEFILE] [--_video _VIDEO]
                         [--_camera _CAMERA] [--_range _RANGE]
                         [--_blockSize1 _BLOCKSIZE1] [--_cValue1 _CVALUE1]
                         [--_blockSize2 _BLOCKSIZE2] [--_cValue2 _CVALUE2]
                         [--_blockSize3 _BLOCKSIZE3] [--_cValue3 _CVALUE3]
                         [--exec EXEC] [--load LOAD]
                         terminal_mode

positional arguments:
  terminal_mode         Flag to run pyforms in terminal mode

optional arguments:
  -h, --help            show this help message and exit
  --_sceneFile _SCENEFILE
                        Scene file
  --_video _VIDEO       Video
  --_camera _CAMERA     Camera
  --_range _RANGE       Frames to analyse
  --_blockSize1 _BLOCKSIZE1
                        Thresh block size
  --_cValue1 _CVALUE1   Thresh C value
  --_blockSize2 _BLOCKSIZE2
                        Thresh block size
  --_cValue2 _CVALUE2   Thresh C value
  --_blockSize3 _BLOCKSIZE3
                        Thresh block size
  --_cValue3 _CVALUE3   Thresh C value
  --exec EXEC           Function from the application that should be executed.
                        Use | to separate a list of functions.
  --load LOAD           Load a json file containing the pyforms form
                        configuration.
```

Full commands examples:

```
d-track-singlecam terminal_mode --_sceneFile 04Hugo201302211037_Scenario.obj --_video␣
↪04Hugo201302211037MergedEntrada.MP4 --_camera Camera1 --_blockSize1 1001 --_cValue1␣
↪296 --_blockSize2 1001 --_cValue2 297 --_blockSize3 1001 --_cValue3 297 --_range␣
↪13500,105249 --exec execute

d-track-singlecam terminal_mode --_sceneFile 04Hugo201302211037_Scenario.obj --_video␣
↪04Hugo201302211037MergedCascata.MP4 --_camera Camera2 --_blockSize1 1001 --_cValue1␣
↪277 --_blockSize2 1001 --_cValue2 277 --_blockSize3 1001 --_cValue3 277 --_range␣
↪13500,105249 --exec execute

d-track-render terminal_mode --_sceneFile 04Hugo201302211037_Scenario.obj --_video0␣
↪04Hugo201302211037MergedEntrada.MP4 --_video1 04Hugo201302211037MergedCascata.MP4 --
↪_data output/04Hugo201302211037_Scenario_3d_tracking.csv --_outputfile test.avi␣
↪exec execute
```

(continues on next page)

(continued from previous page)

2.3 How to use

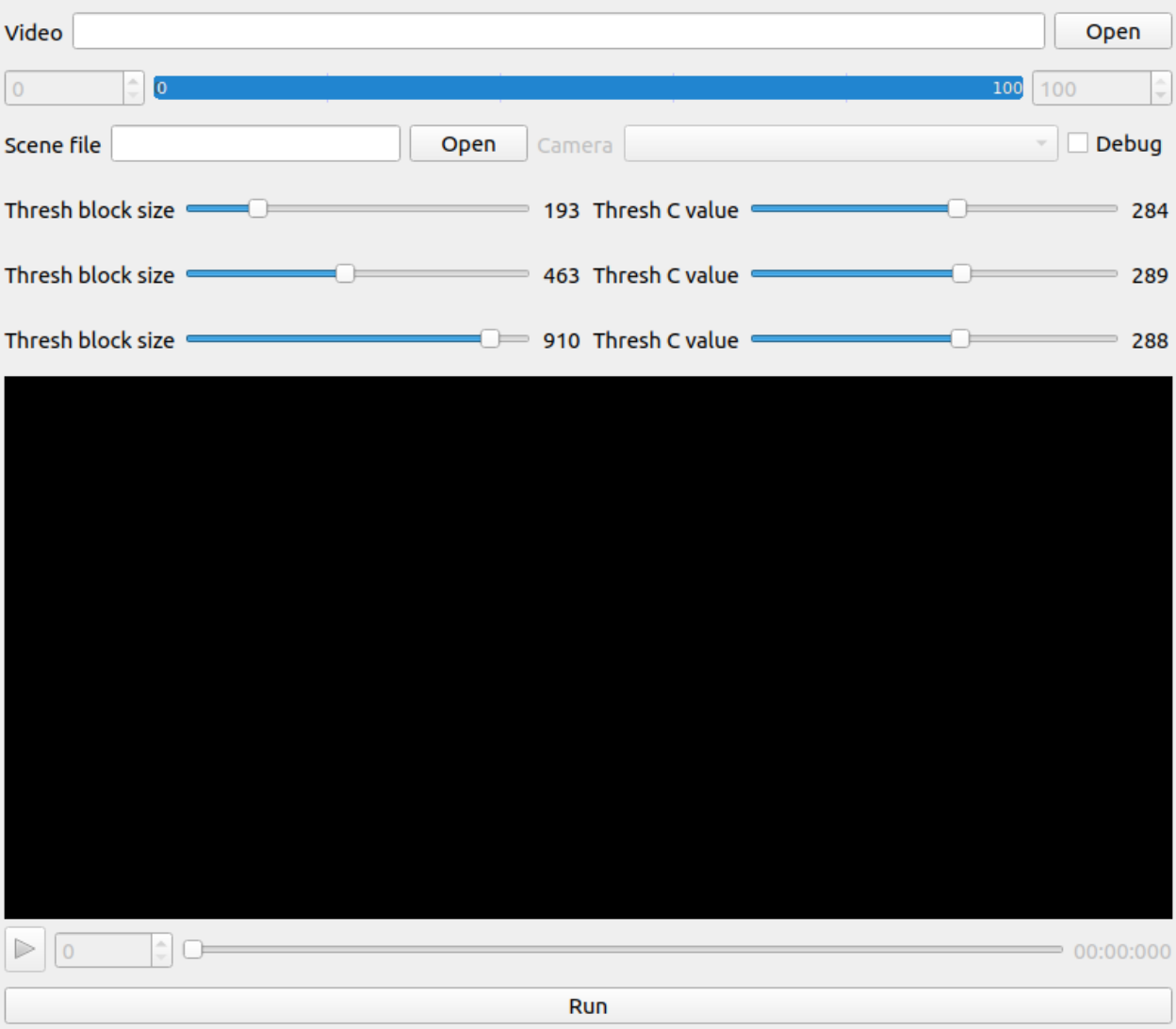
2.3.1 Obtain the 3D path of the dolphin

1. Record 2 sychronized videos of a dolphin swimming in a pool.
2. Create an OBJ file representing the real world scenario with the py3dscene-editor application.
3. Execute the application d-track-singlecam for each of the cameras to extract the dolphin centroid.
4. Use the pythonvideoannotator application to correct any error in the tracking.
5. Execute the application d-track-smoothpath to combine the cameras information and reconstruct the 3D path of the Dolphin.

2.3.2 D-Track

d-track-singlecam

This application is used to segment the images of each camera and find the dolphin in the pool. Select the threshold values for each color channel. Try to color the dolphin as visible as possible.



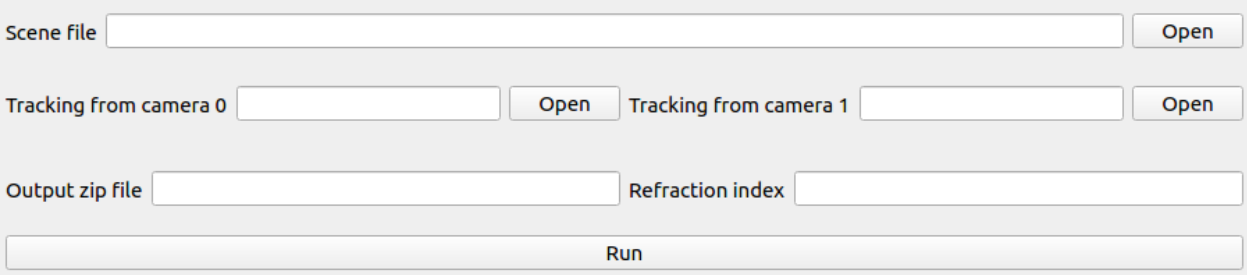
FIELD	DESCRIPTION
Scene file	OBJ file describing the pool. This file is generated by the py3dscene-editor application.
Video	Video file captured with one of the cameras defined in the Scene file.
Frames range	Used to select range of the frames to analyse.
Camera	Name given to the camera on the py3dscene-editor application.
Debug	If active, a video with the results of the colors filtering will be generated.
Thresh block size	Parameter of the cv2.adaptiveThreshold for the blue component of each frame.
Thresh C value	Parameter of the cv2.adaptiveThreshold for the blue component of each frame.
Thresh block size	. . .
Thresh C value	. . .
Thresh block size	. . .
Thresh C value	. . .

Tip: **Output data:** CSV file with the next format.

FIELD	DESCRIPTION
Column 1	Frame index.
Column 2	X pixel coordenate of the blob centroid extracted from the frame after applying the filter 1.
Column 3	Y pixel coordenate of the blob centroid extracted from the frame after applying the filter 1.
Column 4	Area in pixels of the blob extracted from frame after applying the filter 1.
Column 5	X pixel coordenate of the blob centroid extracted from the frame after applying the filter 2.
Column 6	Y pixel coordenate of the blob centroid extracted from the frame after applying the filter 2.
Column 7	Area in pixels of the blob extracted from frame after applying the filter 2.
Column 8	X pixel coordenate of the blob centroid extracted from the frame after applying the filter 3.
Column 9	Y pixel coordenate of the blob centroid extracted from the frame after applying the filter 3.
Column 10	Area in pixels of the blob extracted from frame after applying the filter 3.

d-track-smoothpath

This application is used merge the output of the **d-track-smoothpath** application of the 2 cameras.



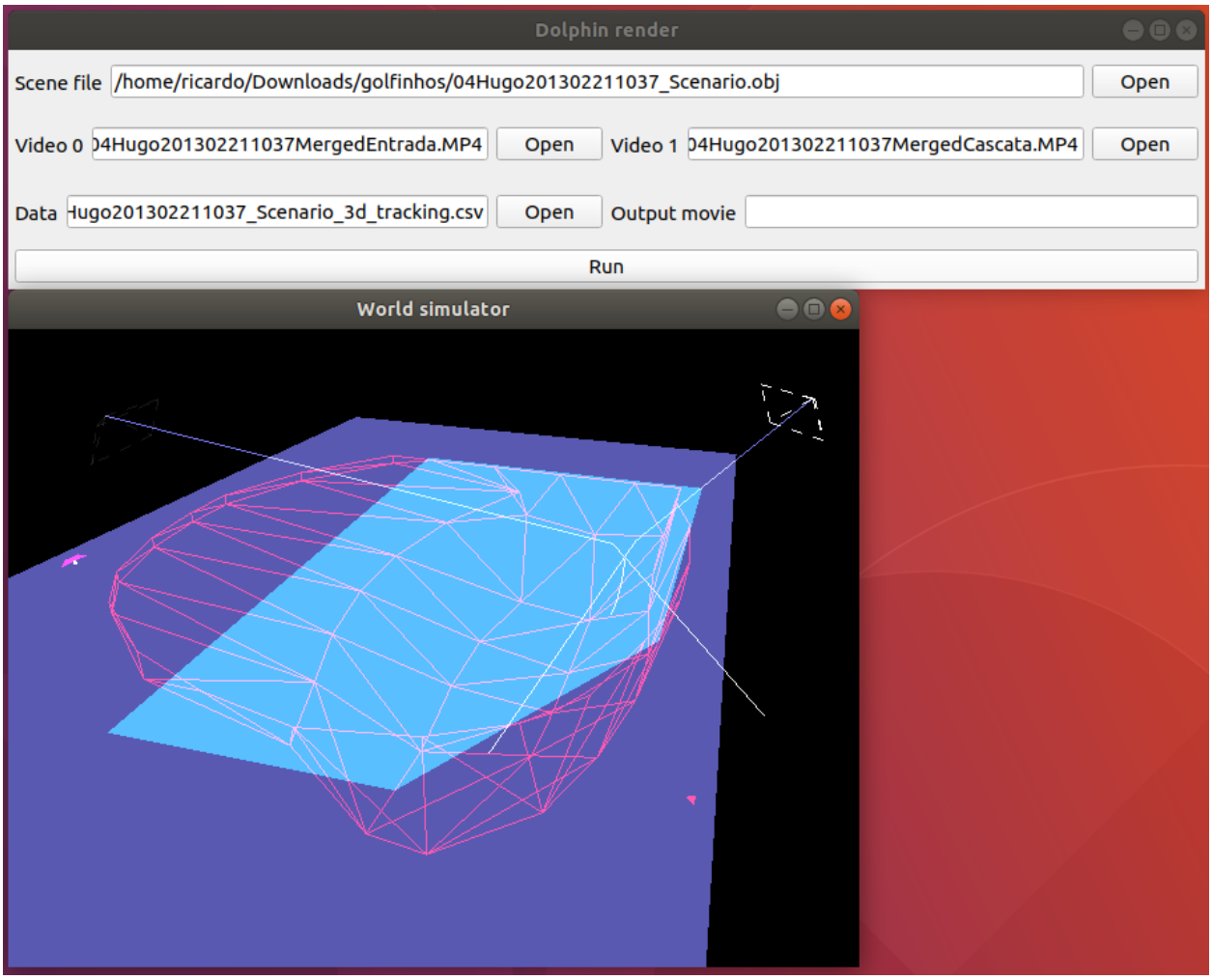
FIELD	DESCRIPTION
Scene file	OBJ file describing the pool. This file is generated by the py3dscene-editor application.
Tracking from camera 0	Camera 1 CSV output file from the d-track-singlecam.
Tracking from camera 1	Camera 2 CSV output file from the d-track-singlecam.
Output zip file	Name of the output zip file.
Refraction index	Refraction index to use to calculate the 3D position.

Tip: **Output data:** CSV file with the next format.

FIELD	DESCRIPTION
Column 1	Frame index.
Column 2	X 3d coordenate.
Column 3	Y 3d coordenate.
Column 4	Z 3d coordenate.
Column 5	Used X pixel coordenate from camera 1.
Column 6	Used Y pixel coordenate from camera 1.
Column 7	The pixel from camera 1 is an estimation (False) or not (True).
Column 8	Used X pixel coordenate from camera 2.
Column 9	Used Y pixel coordenate from camera 2.
Column 10	The pixel from camera 2 is an estimation (False) or not (True).

d-track-render

This application is used render the 3d positions found with the **d-track-smoothpath** application.



FIELD	DESCRIPTION
Scene file	OBJ file describing the pool. This file is generated by the py3dscene-editor application.
Video 0	Camera 1 video file.
Video 1	Camera 2 video file.
Data	File containing the result from the d-track-smoothpath.
Output movie	(optional) Name of a video file where the rendering will be saved. The file should be from avi extension.

Tip: Output data: Video from the 3d rendered scene.

2.4 Example

Note:

Download the files for this example here.

This example execute the applications in terminal mode.

- First use the command **d-track-singlecam** to extract from each camera the dolphin positions pixel positions.

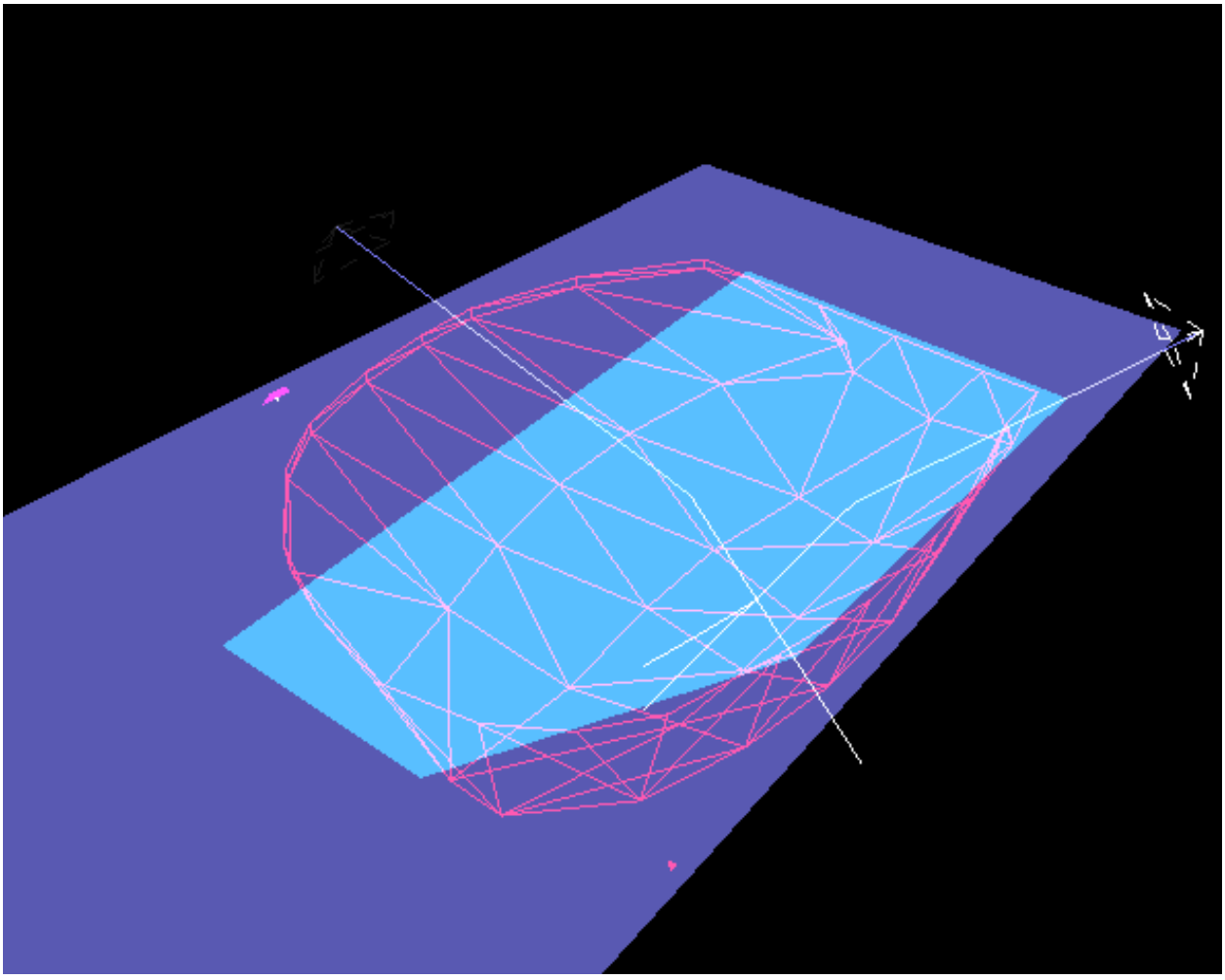
```
d-track-singlecam terminal_mode --_sceneFile 04Hugo201302211037_Scenario.obj --_video␣
↪camera1.mp4 --_camera Camera1 --_blockSize1 1001 --_cValue1 296 --_blockSize2 1001 -
↪-_cValue2 297 --_blockSize3 1001 --_cValue3 297 --_range 0,17965 --exec execute

d-track-singlecam terminal_mode --_sceneFile 04Hugo201302211037_Scenario.obj --_video␣
↪camera2.mp4 --_camera Camera2 --_blockSize1 1001 --_cValue1 277 --_blockSize2 1001 -
↪-_cValue2 277 --_blockSize3 1001 --_cValue3 277 --_range 0,17965 --exec execute
```

- Use the **d-track-smothpath** to combine both cameras pixels positions and estimate the dolphin 3d position.

```
d-track-smoothpath terminal_mode --_scenefile 04Hugo201302211037_Scenario.obj --_
↪trackfile0 output/camera1_out.csv --_trackfile1 output/camera2_out.csv --_
↪refraction_index 1.4 --exec execute
```

- Render the result with the **d-track-render** command.

```
d-track-render terminal_mode --_sceneFile 04Hugo201302211037_Scenario.obj --_video0␣
↪camera1.mp4 --_video1 camera2.mp4 --_data output/04Hugo201302211037_Scenario_3d_
↪tracking.csv --_outputfile test.avi --exec execute
```



Example of a video rendered with the software.