Project FoSIBLE
Fostering Social Interactions for a Better Life of the Elderly

Deliverable

D5.5: Prototypes of evaluation tools to test software components and sensors, Delivery date: M18

Responsible

AIT (Lead Contractor)

Participants

AIT

Version : 1.0
Date: 28.10.2011
Dissemination level: (PU, PP, RE, CO): PU

Abstract

D5.5 describes the prototype evaluation tools to test the software for the gesture recognition interface in the FoSIBLE project. The tools contain recording software for recording, playback and storing the sensor raw data and a front-end to be used for in the project for evaluation and online demonstration of the gesture recognition algorithms.

Table of Content

1 Introduction

1.1 Purpose of the Document

The purpose of this document is to describe the prototype evaluation tools to test software components for the gesture recognition interface in the FoSIBLE project. The tools contain recording software for recording, playback and storing the sensor raw data and a front-end to be used for in the project for evaluation and online demonstration of the gesture recognition algorithms. The concept, functionality, handling and the implementation of the tools are described in detail.

1.2 Definitions, Acronyms and Abbreviations

Acronym	Description
FoSIBLE	Fostering Social Interaction for the Well-Being of the Elderly
IF	Interface
PC	Personal Computer
AE(R)	Address Event (Representation)
3D Sensor	Devices that delivers spatial (three dimensional 3D) information of a given scene , typical x,y and depth-z
SW	Software
HW	Hardware
TAE	Timed address event data
OS	Operating System

2 Overview and Concepts

The following chapters describe the tools for data recording and the evaluation of the gesture recognition interface and how these tools work together in the development process.

The concept of the development work is to record a large number raw UCOS2-XL sensor [1] data of training and test data sets necessary for the development of the software algorithms. Pre recorded TAE data sets are also needed for the training of the gesture classifier. The recording task is often not performed in laboratory environment but e.g. in elderly homes where different subjects should perform a larger number of gestures, therefore a stable and versatile recording tool is needed. This tool is therefore implemented in C/C++ programming language and runs as a binary executable under Windows OS.

The evaluation of the gesture classification algorithms on the other hand needs to integrate constantly improved and changed software modules which cannot easily be done in C-programming language. This tool is therefore implemented in the Matlab programming environment where changes in the code and subsequent debugging can be easily and interactively performed. This tool allows for a live data connection to the sensor and output results in a “mock-up” on-screen GUI incorporating a simple menu structure. This allows to evaluate the performance of the software by receiving direct feedback on the gesture. This evaluation results is in turn used to constantly improve and update the software algorithms. A simple demonstration is also possible by this tool, with the drawbacks that a Matlab license must be available on the demo PC and slight shortcomings in processing speed, resulting in a noticeable lag between gesture and corresponding reaction on the screen.

The recording and evaluation tools resemble the tool chain used in the project. The evaluation tool acts as a test bench for the gesture recognition development. Figure 1 graphically illustrates the project’s software tool chain.

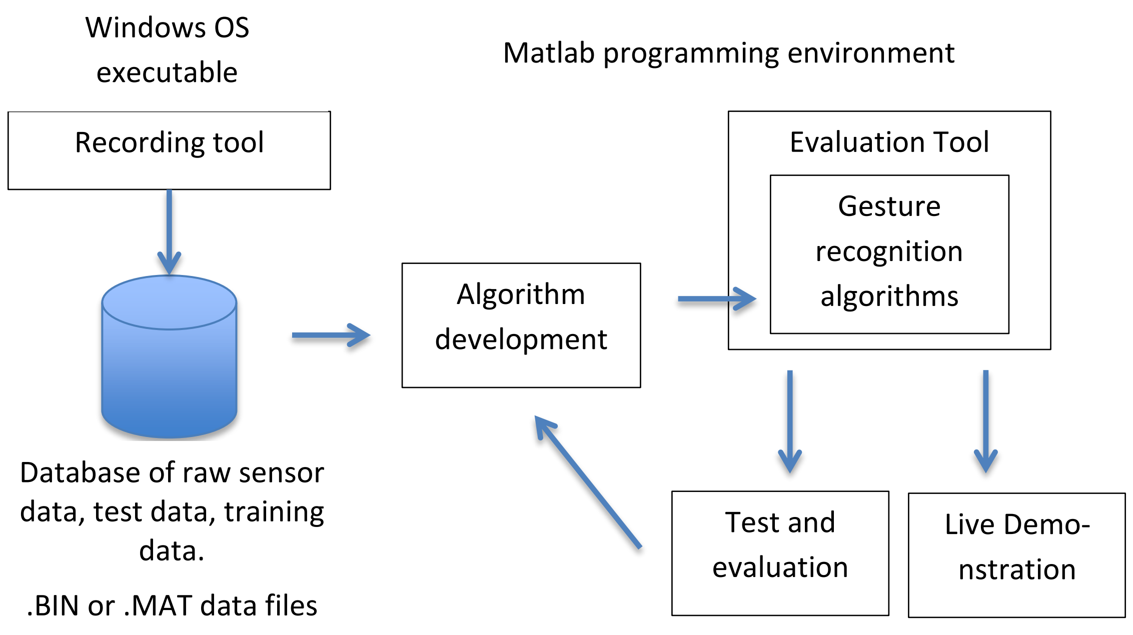


Figure 1 Development tool chain and software tools for the gesture recognition software.

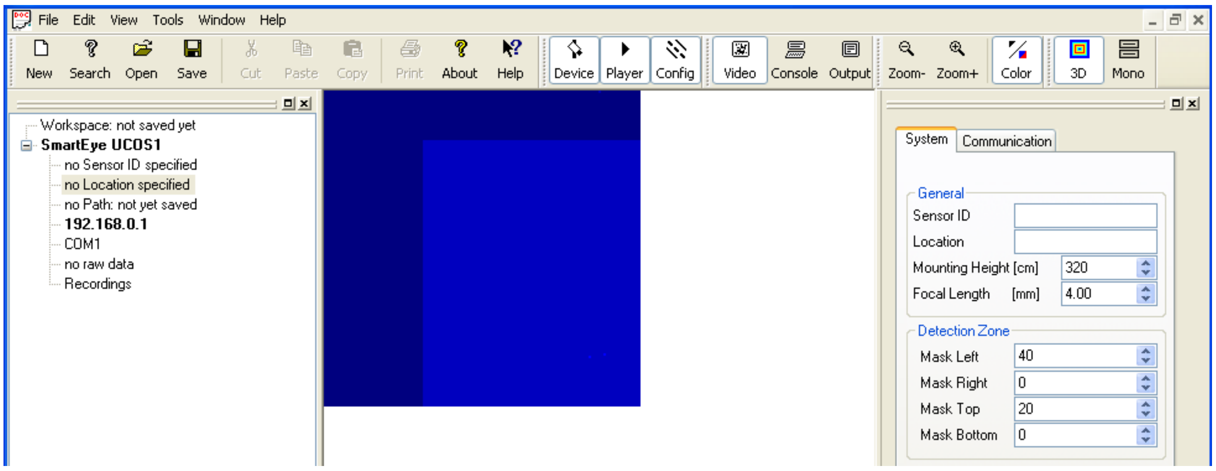
3 Recording Tool

3.1 Overview

The recording tool is a graphical user interface (GUI) for Windows XP and enables the following activities:

- Display of live sensor data
- Configuration of the UCOS2-XL sensor
- Text-based console interface to the sensor (Advanced configuration)

The tool provides a graphical user interface and a text-based console for advanced settings. All sensor settings and connections can be saved and used for maintenance. Figure 2 shows a screen shot of the tool main screen. The window is divided into three sections, the left shows the sensor connection data for establishing Ethernet connection and also for handling test data recordings and files. The center screen is dedicated to different live view modes allowing to show and to control the sensor raw data, as well as the hand tracks needed as the input for the gesture recognition. The right screen part is reserved for sensor settings.


File Edit View Tools Window Help
New Search Open Save Cut Paste Copy Print About Help Device Player Config Video Console Output Zoom- Zoom+ Color 3D Mono
System Communication
General
Detection Zone

Workspace: not saved yet
SmartEye UCOS1
no Sensor ID specified
no Location specified
no Path: not yet saved
192.168.0.1
COM1
no raw data
Recordings

Sensor ID	
Location	
Mounting Height [cm]	320
Focal Length [mm]	4.00
Mask Left	40
Mask Right	0
Mask Top	20
Mask Bottom	0

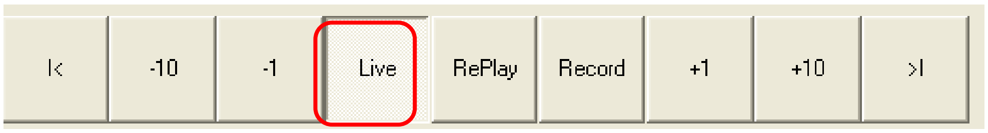
Figure 2. Recording tool screen shot.

3.2 Live-Sensor View

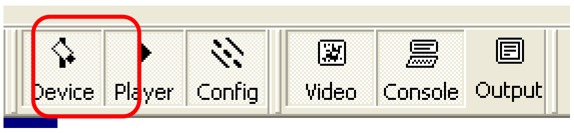
The raw sensor data can be viewed in the live view of the tool.

Set up an Ethernet connection between UCOS2XL and the tool using the menu on left side of the screen. The confirmation of successful connection is the welcome message in the console window of the workspace.

Activate the Live button and Video button in the workspace.



Live button

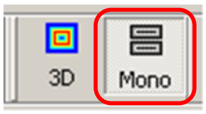


Video button

Mono View

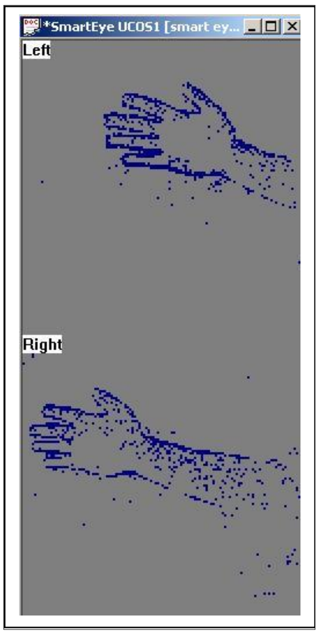
The UCOS2XL is a stereoscopic sensor meaning that the disparity between the left and right sensor image are used to derived the 3D information, i.e. depth from the data. This procedure splits the window for the display of the sensor data into two independent windows, one for the left and one for the right sensor.

Click here in the upper section of the workspace on the **Mono** button.



Mono button

The console displays the command "**stereo off**" and the corresponding response "*Stereo is disabled*". The stereo calculation and the hand tracking mode of the sensor are disabled.

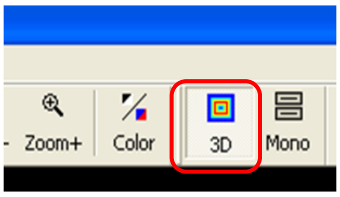


Live Mono View

3D View

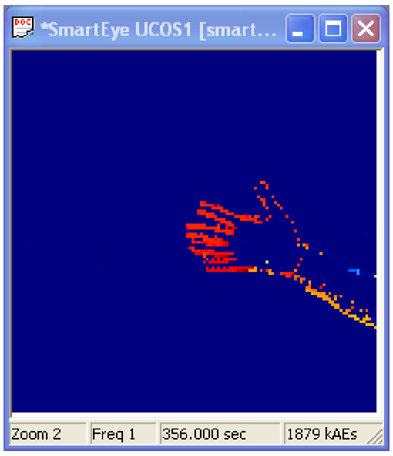
This procedure shows a single output window for the display of the sensor 3D data, which depth information color coded into the image. Blue colors mean that objects are farther away from the sensor, red means that objects are close to the sensor. The sensor will track those objects and is able to output this track data in its data stream.

Click here in the upper section of the workspace on the **3D** button.



3D button

The console displays the command "**stereo on**" and the corresponding response "*Stereo is enabled*". The stereo calculation and the hand tracking mode of the sensor are now enabled and can be viewed live on the screen. The sensor will track the nearest object, i.e. the hand performing the gestures. These Tracks are visible as lines on the screen.



Live 3D View

3.3 Data Recording and Playback

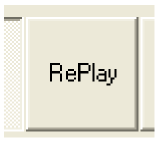
The tools allows recording of raw data while the live view is visible on the screen. For the data recording into data files choose in the left side of the window the menu *Recordings* → *Save raw data as..* → select "bin" or "mat" depending on which file type you want to record. See the following sections for file format details.

To start recording the live data into a file press the *Record* button on the lower edge of the tools main window:

Record button

To inspect the quality and integrity of a recording pre-recorded data, files can be played back with the tool. This is especially important when recordings are made with real subjects and can be only repeated within a short time on-site. For the playback select a playback file source, choose in the left side of the window the menu *Recordings* → *Import raw ..* → select a file in the file browser depending on which file you want to replay.

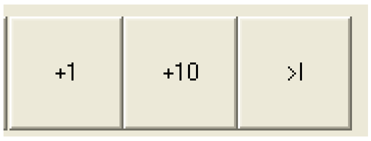
To start to replay the data press the *RePlay* button on the lower edge of the tools main window:



Replay button

For the replay different replay speeds can be selected in the lower left corner of the window by setting the *frame step* and *delay* values there. *Frame step* controls how many frames are skipped in the replay process, e.g. “1” corresponds to real-time replay, “2” replays at double speed. *Delay* controls how much pause is inserted between frames in the replay process, “0” corresponds to real-time replay, “10ms” inserts a hold of 10ms between successive frames.

Frame by frame stepping through raw data recordings is also possible to inspect the data closely if necessary. The buttons +1,-1 step on frame forward or backward, +10, -10 step 10 frames forward or backward, respectively. The buttons |< and >| jump to the begin or end of the recorded file.



Buttons to control frame-by-frame stepping.

3.4 Data Storing and File Formats

The raw data files are created automatically by the tool when a recording is started. The file names are automatically generated from the sensor ID settings and the date and time of the recording assuring a unique file name is created.

Two file formats for raw sensor data storing are available:

BIN .. Binary file format for TAE data which saves raw data in a 2 x 32bit format with timestamp information in the first and address (pixel) information in the second word of each event. The time stamp resolution is given in milliseconds.

MAT .. Matlab file format for TAE data which saves raw data in a 2 x 32bit Matlab array with variable name ***ae***, and timestamp information in the second row and address (pixel) information in the first row of the matlab array. The time stamp resolution is given in microseconds.

3.5 Data File Import into Matlab Programming Environment

Import of binary sensor data stored in bin files can be achieved by a Matlab function **ae_bin2mat.m** that loads the data into a Matlab variable.

```
> ae = ae_bin2mat(‘ucos2_201109101_000000.bin’);
```

This function that reads TAE from the binary data file is implemented in Matlab code as shown below:

```
  fid=fopen(fn, 'rb');
    if (fid==-1)
        fprintf('ERROR: Couldn_t open file "%s"\n',fn);
        ae=[];
        return;
    end

    [ae, cnt] = fread(fid, inf, 'uint32=>uint32');
    ae=reshape(ae, 2, cnt/2);

    fclose(fid);
```

After a file open the function reads one block of uint 32bit data from the file and reshapes the data in such way that subsequent events are organized in columns, address data is in the first row and time stamp data in the second row of the matrix. Data is output by the function as a Matlab array ***ae***.

Import of MAT data files can be easily performed by using the Matlab load command to load the data into the Matlab workspace.

```
> load ucos2_201109101_000000;
```

4 Evaluation Tool

4.1 Live sensor data interface

For the evaluation tool a collection of functions for a live data connection to the sensor and the Matlab programming environment has been developed. The live data interface provides connectivity to the TAE data stream that is transferred from the sensor via Ethernet UDP packets. Each UDP packet is received on a dedicated port. The basic IO routines are based on [7].

The first step is to initialize the function by creating the Matlab object and a callback function:

```
addpath([pwd '\aestream\']);
addpath([pwd '\mex\']);

persistent AEDevTool3D;

P = mfilename('fullpath');
[pathv, name, ext] = fileparts(P);
callback = str2func(name);

AEDevTool3D = BuildGUI(callback);
```

The setup of a periodical timer function is necessary to periodically flush event queue:

```
timer( ...
        'Name', 'flushTimer', ...
        'TimerFcn', 'drawnow;', ...
        'ExecutionMode', 'fixedRate', ...
        'Period', 0.33 ...
);

waitfor(AEDevTool3D)
```

The initialization of the data transfer is done by opening the UDP data socket and sending a command to the sensor to start the raw data streaming:

```
[success msg] = openAEstream(AEDevTool3D);      % bind sockets
[success msg] = startStreaming(AEDevTool3D);   % request start streaming

start(timerfind('Name','flushTimer'));
```

As long as the Matlab object returns that it is in the streaming mode, i.e. receiving data by the sensor, the software periodically polls for data via the **receivePacket** function. If the poll results in useful data, data is accumulated into a Buffer until a data rate threshold is under run signalling that a motion, potentially representing a hand gesture, has ended. This then triggers the call to the gesture recognition module that returns a recognized gesture value or an empty value if no gesture could be extracted from the data. If a gesture is successfully recognized its code/number is passed to the GUI function and the display of the GUI is updated accordingly. After that the Buffer is cleared to acquire new data from the sensor.

```
while(getappdata(AEDevTool3D,'StreamingFlag'))
   [drcvd, bin_value] = receivePacket(AEDevTool3D);

   if(drcvd)

     Buffer = [Buffer bin_value];
     Datarate = mean(diff(bin_value(2,:)));

     if Datarate < thrRate & length(Buffer) > thrLen

     % ******************************************
     % call to gesture recognition algorithm
     % ******************************************
     GestureID = GestureRec(Buffer);

     % ******************************************
     % call to display results on demo GUI
     % ******************************************
     if ~isnan(GestureID)
        GUIupdate(GestureID);
     end

     Buffer=[]; % clear Buffer
     end
   end
end
```

4.2 Gesture recognition

Further processing of the acquired data for the gesture recognition task is documented in deliverable FoSIBLE D5.2 [6].

4.3 Evaluation and demo GUI

The software for the evaluation and demo is implemented in Matlab. The GUI of the tool presents a screenshot of the FoSIBLE TV widget application as "mock-up" to enable a more realistic feedback and demonstration of the system. Overlaid on the screenshot are Matlab graphic objects (array of empty rectangles) simulating the menu choices (see Figure 3). These menus can be navigated through, by the hand gestures performed in front of the sensor. The rectangles fill up when they are selected by the user and become transparent if not selected.

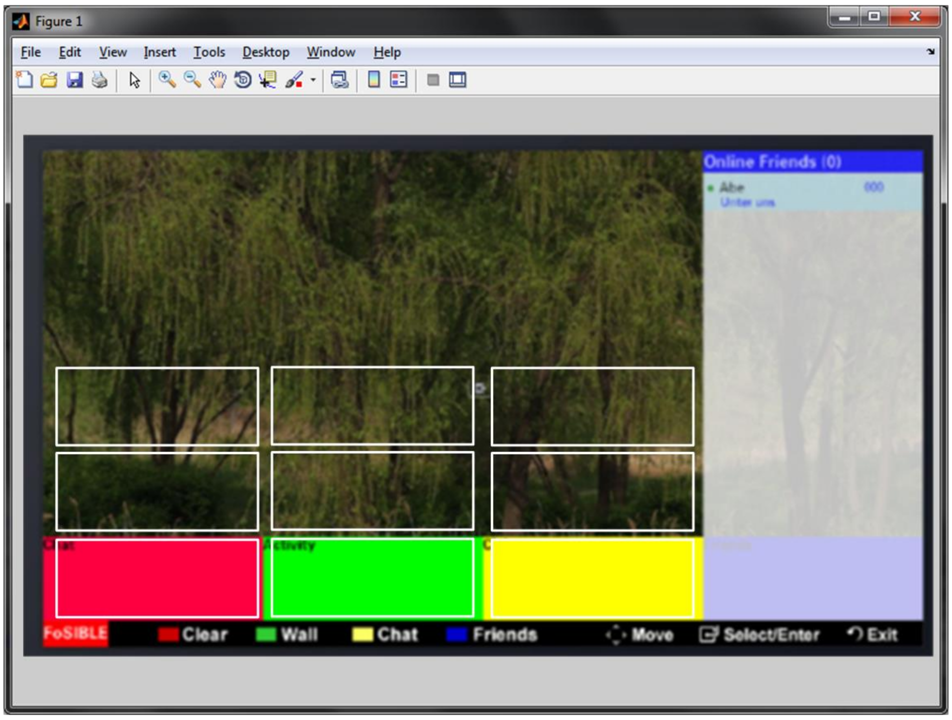


Figure 3. Screenshot of the evaluation and demo GUI running in a Matlab figure window. The background of the window is a snapshot of the actual TV widget.

The concept of navigation through the mock-up menu is shown in Figure 4. Subsequent "up" gestures will move the coloured rectangle upwards in the "array" of possible menu choices.

The GUI tools run in a Matlab figure window with the gesture recognition and the live data acquisition running in the background. Slight shortcomings in processing speed, resulting in a noticeable lag between gesture and corresponding reaction on the screen have to be taken into account in this implementation.

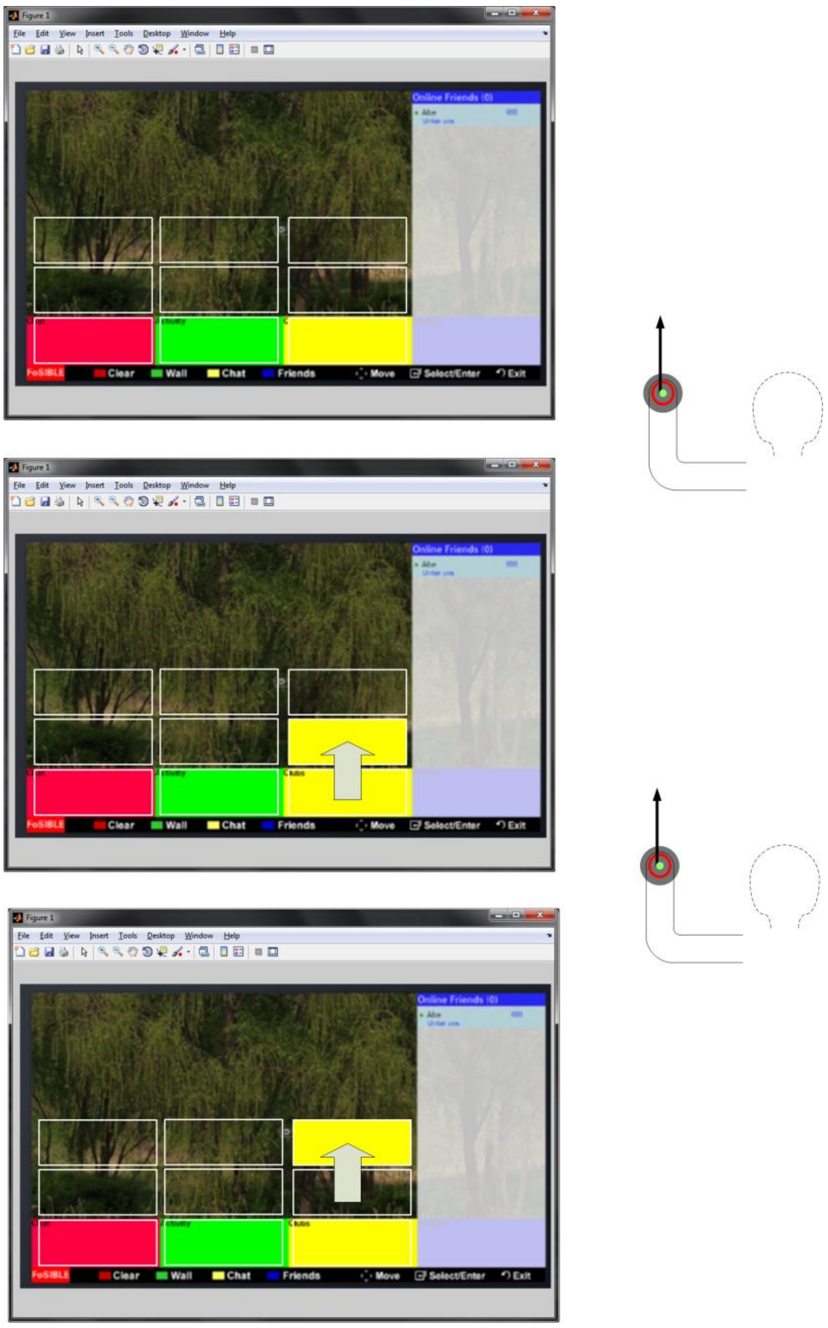


Figure 4. Gestures applied and resulting GUI action in the mock-up menu.

5 References

[1] UCOS2 data sheet - http://www.ait.ac.at/research-services/research-services-safety-security/new-sensor-technologies/development-of-embedded-systems-for-customer-specific-solutions/smart-eye-ucos-universal-counting-sensor/?L=1 as of 2011-08-31

[2] "Lichtsteiner, P.; Posch, C.; Delbruck,; " A 128x128 120dB 30mW asynchronous vision sensor that responds to relative intensity change"; IEEE International Solid-State Circuits Conference, ISSCC 2006; San Francisco; S^3 Digital Publishing, Inc., Lisbon Falls, Maine; ISBN:1-4244-0079-1; pp. 508-509, 669; February, 5.-9., 2006.

[3] Milosevic, N.; Schraml, S.; Schön, P.; "Smartcam for real-time stereo vision – address-event based Stereo Vision"; INSTICC – Inst. f. systems and technologies of information, control & communication; INSTICC Press, Portugal; ISBN: 978-972-8865-74-0; p. 466-471, March, 8-11, 2007

[4] Mead, Carver Silicon retina. In: Mead C, editor. Analog VLSI and neural systems. Reading, Mass: Addison-Wesley, 1989:257–78.

[5] FoSIBLE Deliverable D5.1: "Report on requirements of software components and tools for observation and tracking in laboratory and Living Lab environment." Delivery date M16

[6] FoSIBLE Deliverable D5.2: "Software Prototypes according to D5.1" Delivery date M18

[7] Real-time Fall Detection in Ambient Assisted Living Using a Biologically Inspired Event-based Stereo Vision Sensor, Georg Wiesmann, BSc , Master Thesis, FH Technikum Wien, 28.10.2011