Scene2Model Modelling Tool

Documentation

Christian Muck

Software version: v1.5.0

Contents

1 Abbreviations

- S2M: Scene2Model
- trc: Tag recognition computer
- trs: Tag recognition software
- REST: Representational State Transfer
- virtual machine: vm

2 Introduction to the Scene2Model approach

The aim of the Scene2Model approach is to combine haptic storyboards, made out of paper figures, with conceptual and diagrammatic modelling. The main feature thereby is an automatic transformation from physical representation into a digital model. The haptic storyboards are placed on a table and afterwards an application identifies the used figures and their position and creates a digital representation in a computer-aided modelling tool. The haptic figures and the modelling tool are intended to be used in workshops for creating new products or services. The participants use the paper figures to define and discuss their problems/solutions, by showcasing the customers interaction with new or existing products and services. The automatic generated digital model can than be saved and distributed to different involved stakeholders. It can also be adapted and enriched with additional information.

The used figures are not identified over the presentation of the figures itself, but based on attached tags. These tags are markers combined from different black and white squares. The tags itself are crated with the *ArUco* (cf. `https://www.uco.es/investiga/grupos/ava/node/26` library.

In the context of this service and this document two terms are often used: storyboard and scene. With scene one key moment of a user story is meant, which is visualized by using one setting of haptic figures or one model created with the computer-based tool. A storyboard on the other hand is a composition of one or more scenes to tell a whole story of key moments.

The modelling tool for the *Scene2Model* approach was implemented with the ADOxx modelling platform (`www.adoxx.org`) and is a tool for creating digital storyboards. The representation of the modelling objects and the paper figures are based on the SAP Scenes toolkit (`https://experience.sap.com/designservices/approach/scenes`).

The SAP Scene toolkit consists of paper figures, which can be used to create user stories. The creators of the Scene2Model modelling tool do not possess the rights to the pictures. The rights belong to SAP SE. SAP Scenes^{TM} are distributed under the *Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License*, available under `https://creativecommons.org/licenses/by-nc-sa/4.0/` (last visited at 27. September 2018).

The aim of this documentation is to give a guide in setting up and using the specific functionalities of the *Scene2Model* modelling tool implementation.

Disclaimer: The offered tool is an experimental prototype and no guaranty is given that every offered functionality works completely under any circumstances.

2.1 Structure of the Scene2Model environment

To fully exploit the possibilities of the *Scene2Model* approach a combination of different tools, applications, devices, etc. is needed. This section explains the different parts needed for using the *Scene2Model* to its full potential.

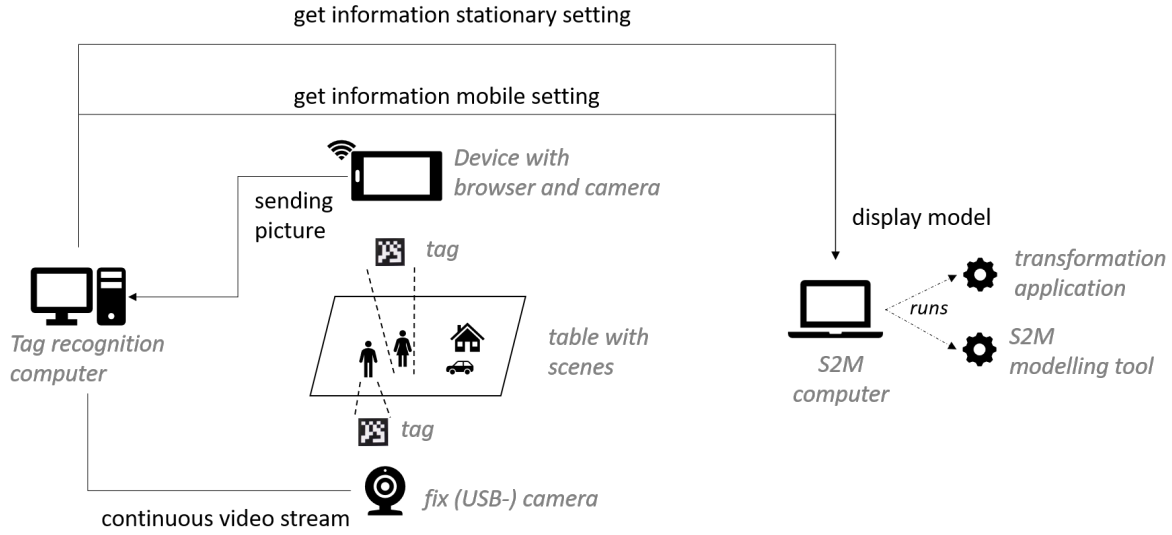

Figure 2.1: Structure overview of a complete Scene2Model installation

An overview of the Scene2Model architecture can be seen in figure 2.1. There are two main parts in the structure of the Scene2Model approach: *Scene2Model (or S2M) computer* and *Tag recognition computer*. The *S2M computer* runs the modelling tool itself and an additional application, which handles the transformation from the tag recognition data to the model and is called *transformation application*. The *Tag recognition computer* copes with the identification of tags in the pictures of the paper figure scenes and offering the data over an interface. There are two possibilities how the *Tag recognition computer (=trc)* can get the pictures in which the tags are identified. One is a continuous video stream from an attached camera and one is a picture sent over a REST interface.

The processing of the video stream offers an continuous stream of data, which is received *transformation application*. If a picture is sent, it is processed and the generated data can be gathered from an REST-resource. The picture can be taken by any device, which can open a compatible browser, has an camera attached and can communicate with the *trc*. More information on the usage of the *Scene2Model* approach with the video stream (cf. section 5.3) and the picture (cf. section 5.2) can be found later in this document.

The tags should face upwards if the picture version is used and downwards for using the video stream.

3 Requirements

In this chapter the requirements for running the Scene2Model modelling tool and/or the tag recognition software are given.

3.1 Scene2Model modelling tool

In this section the requirements for running the modelling tool without the recognition of the marker itself are discussed. The computer must be able to run Windows and the prototype was tested on a laptop with the following hardware specifications:

- processor: Intel® Core™ i7-6700HQ CPU
- speed: 2.6 GHz
- installed RAM: 8 GB
- resolution: 1920 x 1080
- network: WLAN (protocol: 802.11n)

The modelling tool itself was tested on Windows 10 (but is not limited to this Windows version).

3.2 Tag recognition software

The *Tag recognition software* can be downloaded and used in the form of a virtual machine image. It is available in the *.ova* format and therefore, a virtualization application must be used to run the *trs*. It was tested with *VirtualBox* (cf. `https://www.virtualbox.org/`).

The virtual machine can be started on the same device as the modelling tool or on a different one, as long as the modelling tool computer can reach it over an network (e.g. LAN, WLAN, internet, ...). Also, to run the virtual machine the host (computer on which the virtual machine is started) must be able to use virtualization.

The host also needs enough resources to use the virtual machine. In the testing two processor cores and 2048 MB base memory were assigned to the virtual machine. But if available, more resources can be allocated for a better performance. In *VirtualBox* the resources can be configured under *Setting* → *System*.

4 Installation guide

4.1 Install the Scene2Model modelling tool

On the Scene2Model project page, different installation sources can be found. The tool is available for the operating systems Windows, Ubuntu, Fedora and MacOS. For each of them a different .zip folder can be downloaded, containing an installation guide. For Windows the *setup.exe* file must be executed and for the other operating systems installation scripts are available. Under Ubuntu, Fedora and MacOS additional components are downloaded during the execution of the script.

For issues during the installation process, please visit `https://www.adoxx.org/live/installation-guide-15` for detailed instructions and solutions.

During the first start of the modelling tool under Windows, it is possible that the Windows firewall reports that the tool wants to open a connection. This access must be granted if the automatic transformation should be used.

4.2 Install the Tag recognition software

To use the *Tag recognition software*, the virtual machine image must be downloaded from the Scene2Model OMiLAB project web page (`http://austria.omilab.org/psm/content/scene2model/downloadlist?view=downloads`).

Also, *VirtualBox* or another virtualization software, which can read *.ova* images, must be installed on the (physical) device. Because the virtual machine of the *Tag recognition software* was tested with *VirtualBox*, also this documentation will focus on the use with *VirtualBox*.

After *VirtualBox* is installed on the computer, the *Import Appliance* entry in the *File* menu must be clicked. Afterwards, the *omitag_s2m.ova* file must be searched, opened and the rest of the import wizard must be completed. If everything succeeded, the virtual machine can be started. It does not contain a graphical interface, but must be configured, if necessary, via the terminal. Therefore, knowledge in Linux (especially Fedora) are helpful to configure the *trs*.

4.3 Configuration of the set-up

4.3.1 Configure the Scene2Model tool (version 1.5.1

After the installation the tool can be started and used as an stand-alone ADOxx modelling tool. Models can be created, adapted, saved and so on. But if the automatic transformation from the haptic paper figures to the digital model should be used, the tag recognition software and the physical environment must be set up and prepared. Assuming that the tag recognition software is running correctly, the following steps can be used to configure the Scene2Model tool. This description is applicable to the Scene2Model modelling tool version 1.5.1. For the configuration of older tools, please see section 4.3.2.

If you want to configure the Scene2Model modelling tool on Windows, you probably have to start the tool with administrative rights. Therefore, right-click on the icon and choose *Run as administrator*. You may have to confirm that you really want to give this application administrative rights. If you have installed the Scene2Model tool on another operating system, usually no additional rights are needed to use the configuration.

In the started modelling tool you can find the configuration under *Extras* → *Configuration*. After clicking it a window is shown, which informs you that you may need administrative rights to save the changed configuration parameter. This message must be confirmed. The next window shows a list of options. The first entry is *help*, with which an explanation of the other parameters can be shown. The rest of the list contains the different parameters. The entries, which should be shown and/or changed must be marked by clicking (use *ctrl + click* to select more than one). If everything is marked, the window must be confirmed by clicking on *OK*. Afterwards, sequential for each marked entry a separate window is shown. These windows show the current value and allow you to overwrite it. After each of the windows for the different parameters are acknowledged, he last window is shown, where you have to confirm, that the new values are saved.

The following parameters are available:

- serverPort: This is the port, on which the Scene2Model server (usually runs on the same machine as the Scene2Model tool) is listing. This port must be changed, if the default port (2222) is already used on your computer.
- stopServerPort: Is the port, on which the local server is waiting for the stop signal. This port (default: 1111) also must be unbound on your computer.
- websocketAdress: Is the URL under which the socket of the tag recognition software is opened. This address depends on the configuration of your environment. How the tag recognition software is configured can be found in 4.3.3.
- mobileSettingAdress: Is the URL under which the information from a mobile setting can be gathered. This address depends on the configuration of the tag recognition software.

- ontologyPath: Is the path to the ontology file, which should be used for the additional knowledge, which is needed for mapping the information from the tag recognition to the model objects.
- serverURL: Is the base URL under which the server, preparing the information for creating the model, can be reached.
- autoStartServ: Parameter for specifying if the Scene2Model server should be started automatically during the start up of the Scene2Model modelling tool. The value 1 should be used if it should automatically started and 0 otherwise.
- tableLenX: The X-length of the table in cm –¿ used for repositioning of the figures
- tableLenY: The y-length of the table in cm –¿ used for repositioning of the figures

4.3.2 Configure the Scene2Model tool (till version 1.5

After the installation the tool can be started and used as an ADOxx modelling tool. But if the automatic transformation from the haptic paper figures to the digital model should be used, the tag recognition software and the physical environment must be set up. Additional information can be found on the OMiLAB project page (`http://austria.omilab.org/psm/content/scene2model/info?view=home`).

Furthermore, the modelling tool needs to be configured, if the automatic transformation should be used. The configuration files can be found in the installation folder of the Scene2Model modelling toolkit. In the standard installation this folder can be found under: `C:\ProgramFiles(x86)\BOC\Scene2Model_ADOxx_SA`. There the Scene2Model folder with the following files can be found:

- config.properties
- setGlobalVar.asc
- s2mOnto.ttl
- scene2model.jar
- stopScene2model.jar

For the transformation of the haptic figures in their digital representation a second application is needed. This application will be called *transformation application* in this documentation. The *transformation application* will be started on the same device as the modelling tool.

During the start of the modelling tool, the information from the *setGlobalVar.asc* file is loaded and the *scene2model.jar* will be started. The *setGlobalVar.asc* file contains four lines with information needed by the ADOxx modelling tool. The two properties *serverPort* and *serverEndPort* are needed, where the first specifies the port on which the automatic started Java application can be called, to gather the information of the paper

figures. The second port is needed for the automatic stopping of the Java application. These two ports must be identical with the ports given in the *config.properties* file, which is described later on. The *serverURL* string specifies under which base URL the REST service from the Java application can be found. It is not needed to change the URL, if the modelling tool is used in its standard installation. Finally, the *autoStartServ* variable can be used to set, if the *transformation application* should be started automatically witht the modelling tool or not. If it should start automatically the value 1 must be entered, otherwise 0 should be chosen.

The defined ports must be changed if they are already used on the computer. If the ports are available, the configuration files must not be adapted. The standard port for the communication between the modelling tool and the additional Java application is *2222*.

The *scene2model.jar* starts the Transformation Application, which manages the information of the paper figures. This application loads the *config.properties* file during its start-up. This file contains the following parameters:

- serverPort: This defines under which port the ADOxx modelling tool can gather the information for the automatic transformation of the paper figures. This must be the same as *serverPort* given in the *setGlobalVar.asc* file.
- stopServerPort: This defines the port, which can be used to automatically stop the Transformation application. This must be the same as *serverEndPort* in the *setGlobalVar.asc* file.
- websocketAdress: This defines the address under which the WebSocket of the *tag recognition software*, for the stationary setting, can be consumed. This port must be adjusted with the installation of *tag recognition software*, which must be set up separately.
- mobileSettingAdress: This defines the address, under which the data of the mobile demonstration scenario, can be gathered. The address and the port can be changed, the rest must stay the same.
- ontologyPath: This defines in which subfolder of the ADOxx installation folder and under which name the ontology is saved.

The *s2mOnto.ttl* file contains an ontology in turtle format. It contains the additional information which is needed to create the modelling objects in the ADOxx modelling tool. The paper figures are recognised over an extra attached marker. For the stationary setting, the marker is placed on the bottom of the figures and faces downwards. For the mobile one, the markers facing upwards. These markers only provide an ID (as integer) and the information on the connected modelling objects is saved in this ontology. The assignment of the tag ID to the modelling objects can be changed in the ontology. The changes will be used after a restart of the additional Java transformation application. A description of the ontology and how it can be adapted can be found in section 4.3.4.

The last file in the folder is called *stopScene2model.jar* and is used to close the Java application, during the closing of the ADOxx modelling tool.

4.3.3 Configure the Tag recognition software

The *Tag recognition software* can be used over its virtual machine (=vm) image and therefore this section describes the configuration of the virtualization software (VirtualBox in our case) and the application (in the virtual machine image) itself.

In VirtualBox two aspect should be checked after the import of the *trs* vm image, aside from the resource allocation. On one hand the network properties and the USB configuration on the other. Both can be checked under the *Settings* button.

Under *Settings* → *Network* → *Advanced* → *Port Forwarding*, the table which is shown in figure 4.1 can be found. It can be used to define how the ports of the virtual machine can be reached on the host device. In the standard configuration, the *trs* opens the port **8080** for the http requests and port **22** for a *Secure Shell (ssh)* connection and on the host device port **8080** must be called for the http requests (and the WebSocket connection) and port **1234** to open a *ssh* connection.

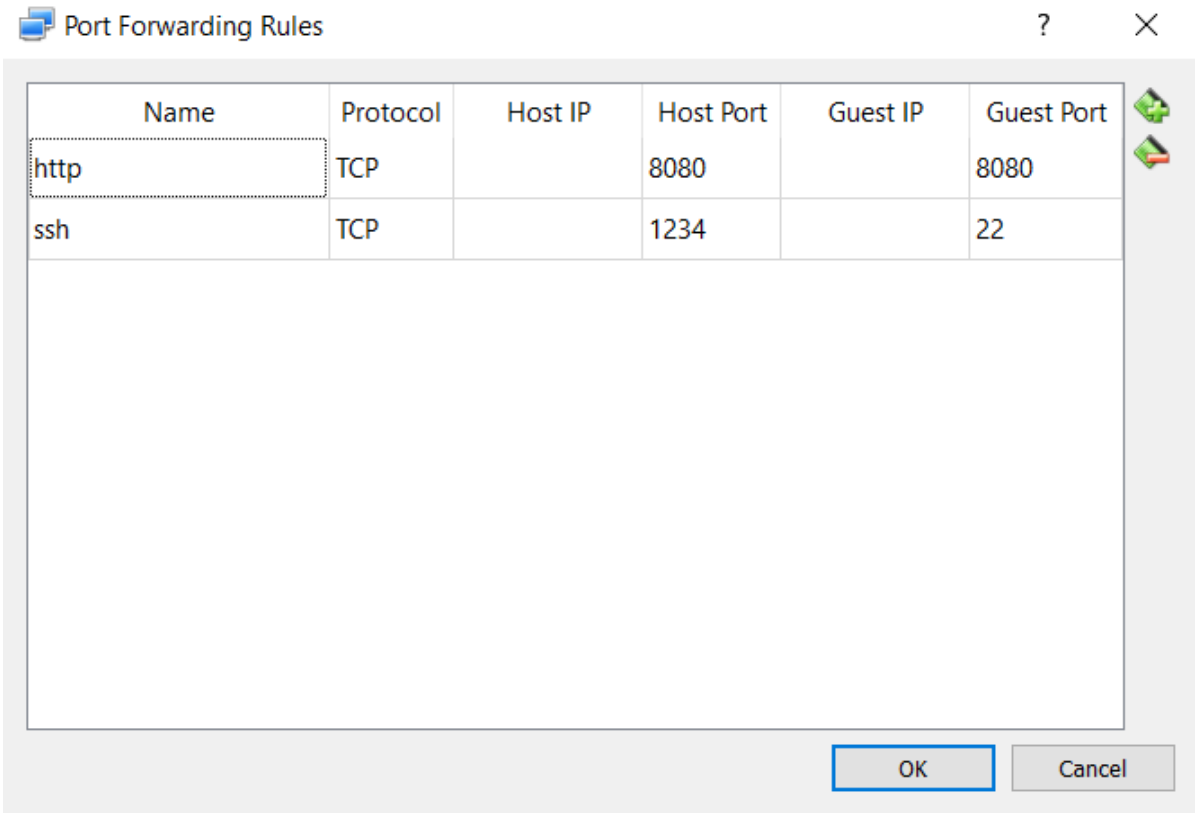

Figure 4.1: Port forwarding table for the virtual machine of the *Tag recognition software*

If no other ports are occupied on the host, the standard configuration (as seen in figure 4.1) can be used. If the standard configuration is used and the modelling tool and the virtual machine are started on the same device, the following configuration in the Scene2Model modelling tool must be used:

- websocketAdress=ws://localhost:8080/omitag/

- mobileSettingAdress=http://localhost:8080/cam/img

More information on how the Scene2Model modelling tool can be configured, can be found in section 4.3.2.

If the stationary setting (cf. section 5.3) should be used, also a camera must be assigned to the virtual machine. The virtual machine was tested with a *Logitech HD pro Webcam C920* USB camera. Under *Settings* → *USB* this configuration can be made. In figure 4.2 the window for the USB configuration can be seen. *Enable USB Controller* (figure 4.2 number 1) must be selected. The virtual machine was tested with using *USB 3.0 (xHCI) controller* (figure 4.2 number 2) was used. Number 3 in the figure shows where a USB device can be added to the virtual machine.

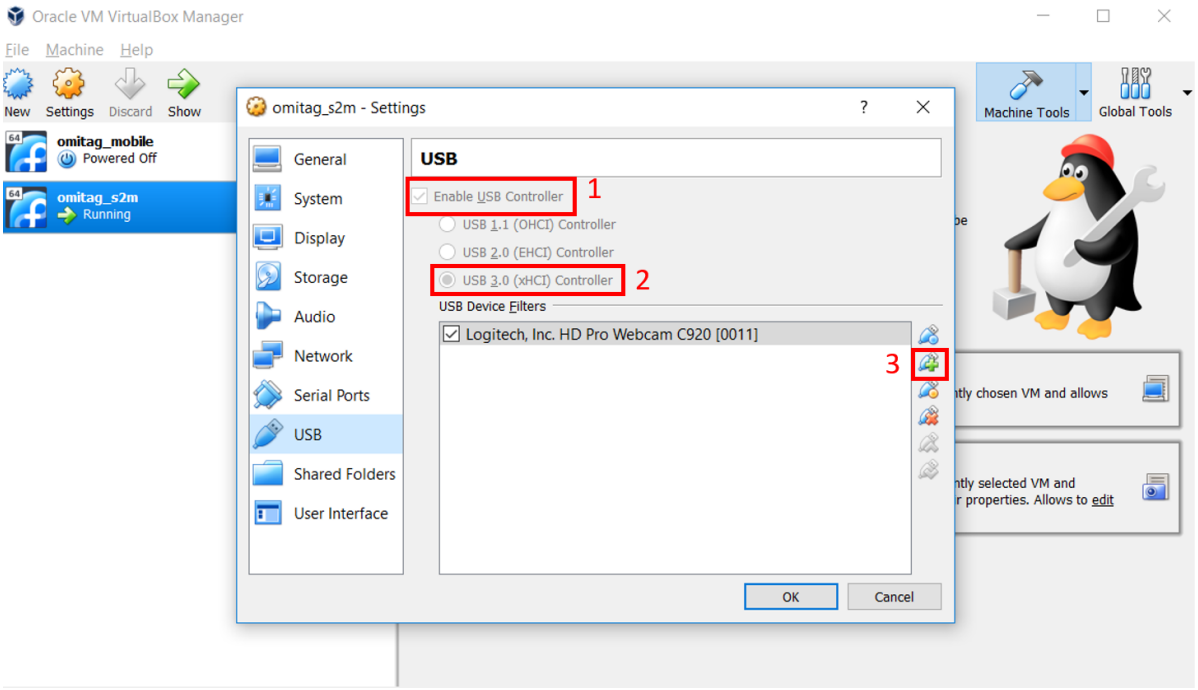

Figure 4.2: USB configuration window for the *Tag recognition software* virtual machine

In the *Tag recognition software* itself also some configuration properties can be adapted. To change these properties the user needs to be able to operate a Linux-based operating system (Fedora in this case) over the terminal. Two users are available on the virtual machine:

- normal:

  username: user

  password: pass

- superuser (root)

  username: root

  password: pass

The application for the tag recognition starts automatically with the start of the virtual machine. The service is called *omitag* and can be controlled with the *systemctl* commands like *status, start, stop, restart, enable and disable*.

The configuration files can be found in the */opt* folder. To change to this folder the ″cd /opt″-command can be used. The *config.properties* file saves the main parameters for the whole application and can be changed using *vi* for example. Consider that root rights are needed to change files in the */opt* folder. In the *config.properties* file, the following important parameters can be found:

- **port**: Port under which the different functionalities can be reached. It is used for the *WebSocket* of the stationary setting and the web page, for taking the picture, of the mobile setting.
- **omitagCmd1**: The first string (until the space) specifies which *omitag* application is started for processing the video stream. The string after the space defines which configuration file should be used for the processing of the video stream.
- **omiphotoCmd**: This string defines, which *omiphto* application should be loaded to process the pictures, which are sent by the mobile setting.

The *omitag1.yml* in the */opt* folder is the standard configuration file for the tag recognition in the video stream. Some of the parameters are only needed if the *Tag recognition software* itself is changed and therefore will be not described. It is possible that the following parameter must be adapted, if the standard version of the *trs* is used. The following important parameters can be found in the *omitag1.yml* file:

- **videoDevice**: specifies which plug-in video device is used for gathering the video stream for the stationary setting (cf. section 5.3). To show the devices, which are currently available on the virtual machine, the command **v4l2-ctl –list-device** can be used. If the command shows the USB camera has another name */dev/video0*, this parameter has to be updated. With the before mentioned command, it can also be checked, if the virtual machine recognised the connected USB camera.
- **points**: This parameter is used to configure the fixed tags for the stationary setting (cf. section 5.3). These tags are important, because they are the basis for calculating the position of the paper figure tags. For each tag, the *id* and its coordinates (x and y) must be specified. The z coordinate must be given, but is not used in the Scene2Model approach. Z would specify the distance from the tag to the wanted zero-level. So in the given tag position, z is always 0, indicating that all tags are on the same level. The coordinates x and y stands for the position on the table. The position is defined in millimetres and the center of the tags, must be placed on the table at the defined position. More information to the fixed tags and their position on the table can be found in section 6.1. If the fix tags are positioned on other coordinates, they new coordinates can be entered under this

parameter and the *trs* can be used as before. The format for defining the fixed position is as following below. The first line shows the format, using *???*, where the values should stand and the next to lines show examples:

- {id: ???, x:???, y:???, z:???}
- {id: 115, x:100, y:100, z:0}
- {id: 23, x:700, y:100, z:0}

4.3.4 Adapting the Scene2Model tool

In this section a short introduction on how the Scene2Model modelling tool can be adapted is given. The adaptations described in this section are focused on how to add new objects to the Scene2Model modelling tool or change existing ones and how the information from the existing tags can be adapted or how new tags can be added. The description starts with explaining how the information from the tags can be adapted and afterwards gives an introduction about adding new objects to the modelling tool.

Adapting the tag information

First of all it should be mentioned, that the tags alone only represent an integer value. The knowledge about which tag represents which object is saved in an ontology. To change the automatic created object in the modelling the ontology must adapted. The ontology file uses the *Turtle* syntax (`https://www.w3.org/TR/turtle/`) and is called *s2mOnto.ttl*. The file can be found in the *Scene2Model* folder in the installation path of the modelling tool. Therefore, if the information for transforming the tags into modelling objects should be altered, knowledge about ontologies and the Turtle syntax is useful.

In the beginning of the ontology file the different classes, their attributes and the values for the different types are defined. Adaptations here are only needed if new types of objects (new graphical representations) or classes should be added. In the ontology classes of objects defined as a *rdfs:Class* and are a *rdfs:subClassOf* of *Sceneobject*. The *rdfs:label* attribute is used to save the name of the class, which must be the same as the class name in ADOxx.

For defining the types in the ontology, the *rdf:Alt* class is used and a list of the different types is given in the ontology. Each list entry again has a *rdfs:label*, which is the same as the name of the type in ADOxx.

If only the connection of tags and existing classes should be made, the before mentioned descriptions are not necessary. Therefore, only the individuals defined at the end of the ontology file are needed. The beginning of the individuals is marked with:

```
#################################################
#
# Individuals
#
```

##################################

For each tag which should be used for the automatic transformation, an individual in the ontology file must be added. For example for an individual can be seen here:

```
:ch_business_man a :Character ;
        :hasTagID 25;
        :hasName "Sebastian Huber"@en;
        :hasDescription "Sebastion is a manager in a huge company."@en;
        :hasRole "manager"@en;
        :hasAge 35 ;
        :hasView :front ;
        :hasType :Business_Man.
```

This example shows how the information is stored to turn the tag with the number 25 into a *Character* modelling object with the type *Business_man*. It is important that the individual posses a *a*-property to a *rdfs:subClassOf* a *:SceneObject*, a *:hasTagID*-property to the number of the tag and a *:hasType* to the type, available in the modelling object. Over the other properties values for the other attributes of an object can be set.

Adapting the modelling tool

For changing the objects which are available in the Scene2Model modelling tool, knowledge of the ADOxx metamodelling platform (www.adoxx.org) is necessary. The S2M modelling tool was developed using ADOxx and therefore, can also be changed using this platform.

The ADOxx platform, consists of two applications: *ADOxx Development Toolkit* and *ADOxx Modelling Toolkit*. The first is used to create the modelling method/tool and therefore the available objects and their attributes can be defined here. The *modelling toolkit* can than be used to create models.

To adapt and use the new version of the Scene2Model tool, it has to be prepared in the ADOxx tool. Therefore, the Scene2Model .abl file (available at its OMiLAB project page) must be imported in the *ADOxx Development Toolkit*. Further, the folder *Scene2Model* (also available at its OMiLAB project page) must be copied into the ADOxx installation folder. The configuration is than similar to the one described in section 4.3.2, with the only difference that the configuration files must be searched in the ADOxx installation folder and not in the Scene2Model installation folder.

In the *ADOxx development toolkit* the created classes can be found. The classes for creating the scenes are all inheritors of *_SceneObject_*. All classes which show text in the canvas of the modelling tool, e.g. speech bubble, tablet, etc., are also inheritors from *_SceneTextClass_*.

If one wants to add a new class of objects to the modelling tool, than the new class should inherit from one of the before mentioned classes. If this inheritance is defined from *_SceneObject_*, the new class posses the following attributes:

- **Name**: String for giving each object an unique name.
- **Type**: Enumeration for defining the different types, a class can have. Based on the type the graphical representation can be defined.
- **Description**: String for giving each object a description.

If the new class inherits from the *_SceneTextClass_*, also the *Text* attribute is available, which can be used to show text in the graphical representation of an object.

Further, the new class may needs to be prepared for showing licence information. Therefore, an *Expression* attribute is used in the Scene2Model tool, which get the needed licence title and information, depending on the chosen type. An example, can be found in the already given classes of the Scene2Model ADOxx library.

For defining the graphical representation of a new class or an existing class, first the picture has to added to the database, using the *Extras* $\rightarrow$ *File management* in the opening window of the *ADOxx Development Toolkit*. It is important that the picture is added to the right library folder. Afterwards, the *Type* attribute of the class, with the new graphical representation must be extended with the new Type. Afterwards, the *GraphRep* attribute of the class must be adapted to show the new picture with the new type.

5 Using the modelling tool

Because the modelling tool was developed using the ADOxx metamodelling platform, it also posses the standard functionalities of it. For example, queries about the models can be asked, a file export and import can be done, pictures of the models can be exported and so on. In addition to the standard functionality also specific ones were implemented, which will be described in this chapter.

5.1 General functionality

5.1.1 Creation of a Storyboard/Process Map

The modelling tool offers the functionality to automatically create storyboard models or process map models. Storyboards consists of objects which contain links to different scene models. In a process map, the processes which used in the different scenes are shown in one overview model.

The logic for both functionalities are equal. A scene model must be opened to enable the triggering of the automatic creation. Afterwards, scene models in the same *Model group* (or folder) are taken in alphabetical order and are processed subsequently. Therefore it is advised, that the scene models which belong together are saved in one *Model group* and their order is defined by starting their name with a number.

The functions can be found in the *Scene2Model* menu item (cf. figure 5.1, number 3 and 4).

5.1.2 Starting/Stopping the transformation application

The *Transformation Application*, handling the transformation from the haptic scenes to the digital model, can be started and stopped within the modelling tool. Therefore, two entries in the menu entry *Extras* are available. The entries are called: *start S2M server* and *stop S2M server*.

Further, the *Scene2Model* modelling tool, can be configured, to start the transformation application automatically or not. Therefore, a value in the *setGlobalVar.asc* file from the sub-folder *Scene2Model* of the Scene2Model tool installation folder must be changed. In this file a variable (line) with the name *autoStartServ* can be found. Its value determines the automatic start of the transformation application. If it should start automatically the value 1 must be entered, otherwise 0 should be chosen.

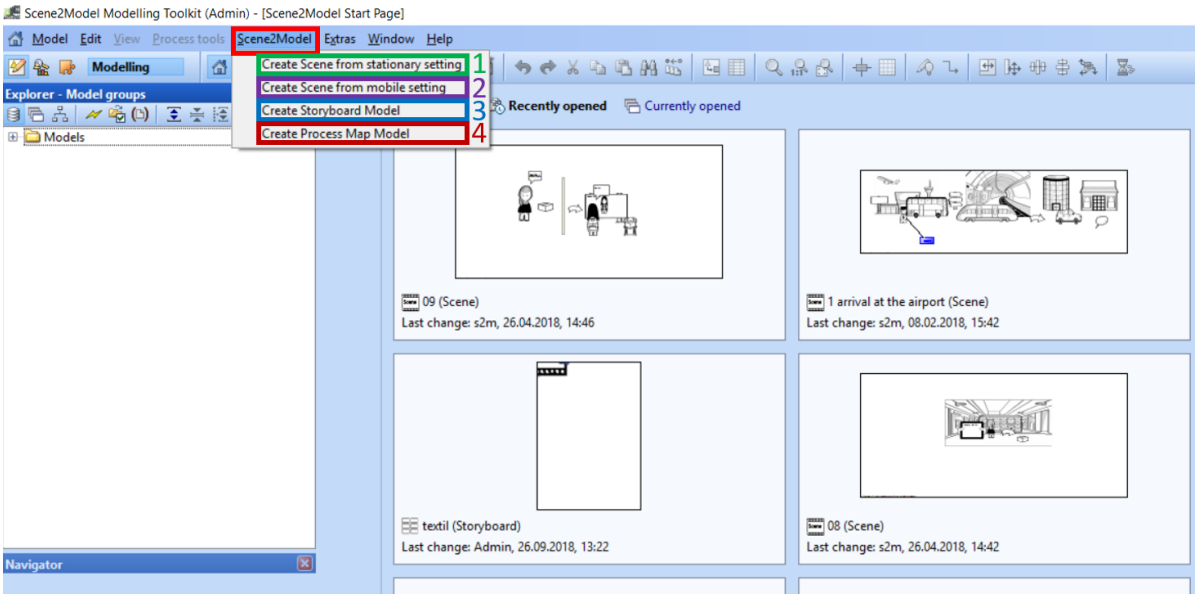

Figure 5.1: Scene2Model interface with the *Scene2Model* menu entries

5.2 Automatic transformation in the mobile setting

For the *Scene2Model* modelling tool, a mobile scenario for demonstration purposes is available. It allows users to take a photo of paper figures with tags (facing upwards) and transform it into a digital and diagrammatic model of the size of an A4 sheet. The size is chosen, because the paper figures should also be placed on a given A4 paper sheet. To use this functionality, the Scene2Model modelling tool must be downloaded and installed and in addition the *Tag recognition software* must be started. More information on the *Tag recognition software* can be found in section 4.3.3.

The first time, the automatic mobile transformation is used, the Scene2Model modelling tool must be configured to allow a connection to the *trs*. How the configuration is done can be found in section 4.3.2. If the *trs* is started on the same device as the modelling tool and the standard configuration is used, than the following parameter must be entered in the *config.properties* file of the Scene2Model modelling tool:

- mobileSettingAdress=`http://localhost:8080/cam/img`

After the configuration is done and both applications are started. The website of the *trs* for taking the picture must be opened. How it can be opened depends on the set-up. If the web page (for taking the picture) will be opened on the same device as the *trs* and the standard configuration is used, than the following URL must be typed into the browser:

- `localhost:8080/cam.html`

If the picture web page is opened on another device, the computer running the *trs* must be reachable over the internet or a local network. Further the IP-address, of the

device running the *trs* must be used instead of *localhost* in the URL. For example, a smart phone can be used to take the picture. If the picture web page is reached, but no window with a camera is shown another browser should be tried. For example, *Google Chrome* don't allows the opening of a device.

In figure 5.2 a screen shot of the web page for taking the picture is shown, after a picture was taken. Number 1 shows the video stream from the opened camera. Under number 2 the taken picture is shown and the button with the number 3 is used to take the picture. Number 4 shows the generated number to identify the taken picture.

After the picture is taken, the taken picture (cf. figure 5.2 number 2) and a number above (cf. figure 5.2 number4) it are shown on the web page. It can take some time, till the number is shown, depending on the network connection and the computing power of the device running the *trs* or taking the picture.

After the number is shown the automatic transformation in the *Scene2Model* tool must be started. Before the transformation can be done, it should be controlled if the *transformation application* is running. It is shown in a separate window in the taskbar of the Windows computer (cf. figure 5.3). If this window is not shown, it can be started under *Extras rightarrow start S2M server*. Also, a *Scene* model must be opened. Beware that objects already existing in the *Scene* model will be deleted, before the new model is created. The next step is to click *Scene2Model* $\rightarrow$ *Create Scene from mobile setting* in the menu bar (cf. figure 5.1 number 2). A window will open, in which the number from the picture web page must be entered and confirmed with clicking on *Apply*. If everything was configured right the model objects should now be shown in the drawing area of the *Scene2Model* modelling tool.

5.3 Stationary setting

For using the stationary setting a table with a transparent table top must be prepared. The idea is to let the tags face downwards and place the camera beneath the table top. The big difference to the mobile setting is, that a video stream is used to capture the used tags and not a picture. The advantage is that there is no need for taking a separate picture for every scene that should be imported. Only one button must be clicked in the *Scene2Model* modelling tool to automatic create a scene model in the modelling tool. The disadvantage here is that the set-up of the environment is more difficult than in the mobile setting.

First the table and the camera must be prepared. How this is down in detail will be discussed in section 6. After the table and the camera are mounted the camera must be connected to the computer running the *Tag recognition software*. If the virtual machine is used the USB camera must be configured in the virtualization software. How this can be done with *VirtualBox* can be found in section 4.3.3.

The URLs where the *trs* can be reached, depend on the set-up. The rest of the description assumes that it is started on the same device as the modelling tool and with the standard configuration. If another set-up is used it is possible that an IP or web address instead of *localhost* or another port must be used.

If the standard configuration on the same device is used, the following entry must be used in the *config.properties* file in the *Scene2Model* tool:

- websocketAdress=`ws://localhost:8080/omitag/`

More information on how to configure the *Scene2Model* tool can be found in section 4.3.2.

If the virtual machine is started and proper configured, the web page `localhost:8080` can be reached on the same device. It will show the video stream of the camera, so that it can be controlled, what is seen by the camera. For the recognition it is important, that at least 4 of the fixed tags are seen and recognised by the camera. If the tag is recognised, a cube and the tag id are painted onto the tag.

For the automatic transformation a *Scene* model must be opened in the *Scene2Model* modelling tool. Before the transformation can be done, it should be controlled if the *transformation application* is running. It is shown in a separate window in the task bar of the Windows computer (cf. figure 5.3). If this window is not shown, it can be started under *Extras rightarrow start S2M server*. Be aware that existing model objects are deleted before a the new objects are created automatically. By clicking *Scene2Model* $\rightarrow$ *Create Scene from stationary setting* the automatic transformation is triggered and the created model should be shown in the drawing area of the *Scene2Model* modelling tool.

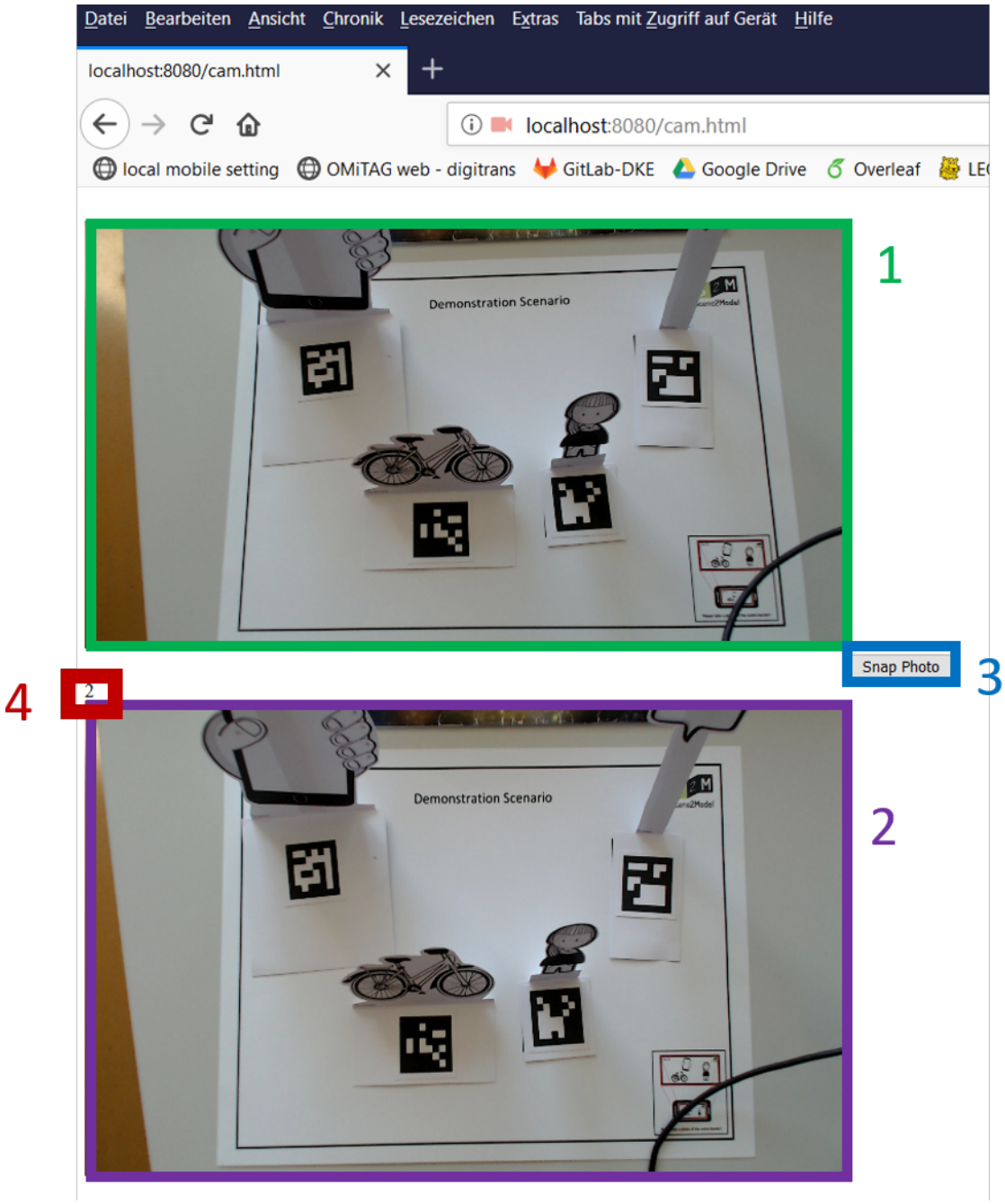

Figure 5.2: Screen shot of the web page for taking the picture in the mobile setting

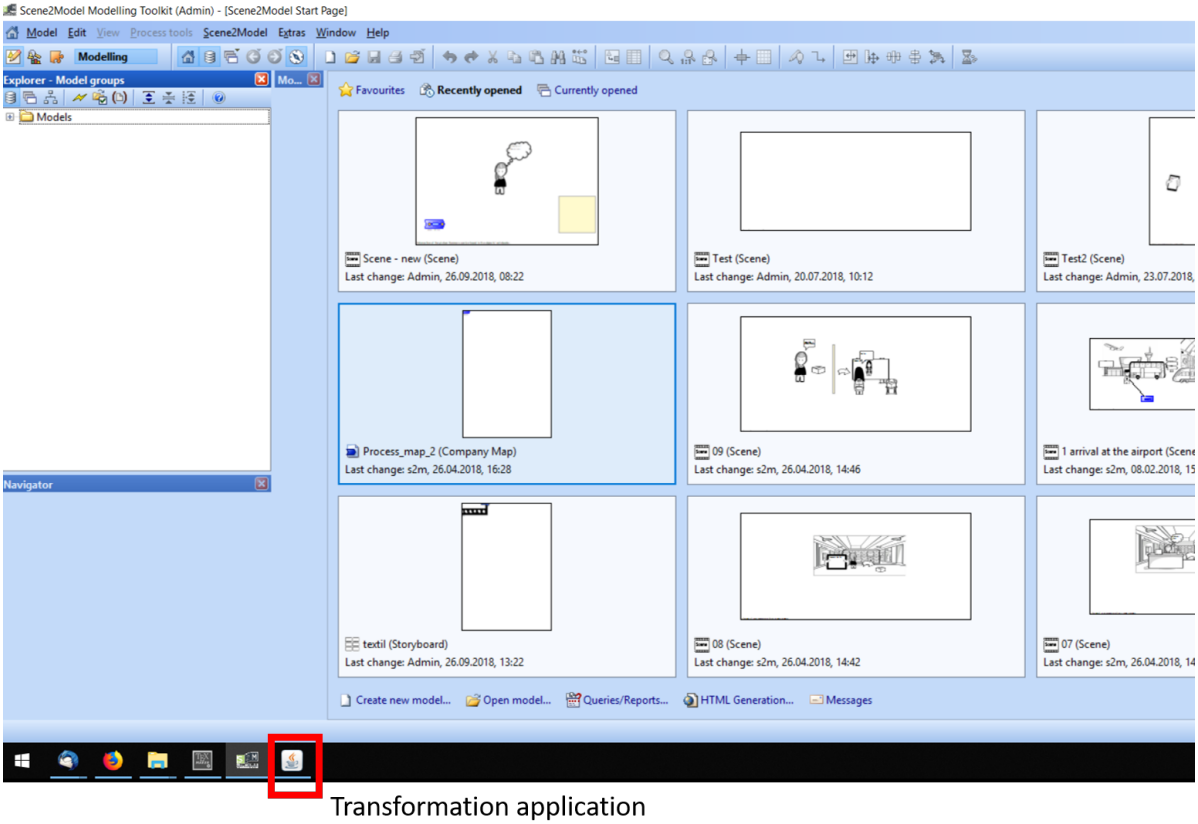

Figure 5.3: *Scene2Model* screenshot with marking the *transformation application* windows

6 Setting up the physical environment

Even though the *Scene2Model* application can be used as a modelling tool alone, it was created to combine physical with digital modelling. Therefore, to explore the overall functionality of the *Scene2Model* approach, also physical elements are needed, which will be described in this section. These physical elements are especially needed if the stationary setting is used.

6.1 Table (stationary setting)

For the stationary setting two variations are possible. The first needs a table with a transparent table top. In this way the camera can be positioned under the table and also the tags can face downwards. The upper side of the table can be used for creating the scenes. By the second variation the camera is placed above the figures and a paper with the fixated reference tags is placed on the table. The reference tags and the one from the figures are always facing in the same direction, towards the camera (whatever the camera is above or beneath the table top). Further the camera must always see at least four reference tags. If the view is disrupted for a longer time period, the Scene2Model server will loose the *WebSocket* connection and must be restarted. How it can be restarted can seen in section 5.1.2. The A3 paper sheet with the reference tags for using it with the variation, where the camera is above the table top, can be found under `http://vienna.omilab.org/repo/files/S2M/Scene2Model%20Reference%20Tags.pdf`. If the reference tag form the sheet are printed on A4, it must be secured that the tags and their distance to each other stays the same.

The reference tags from the paper sheet and from the first variation (camera beneath the transparent table top) can not be used together. If the camera sees tags from both variations, the calculated corrdination of the paper figures will not represent their true position.

The relative position of the camera to the table top is not that important, because as long as four fix tags and the tags of the paper figures are seen by the camera,the figures and their position can be recognised. Nevertheless, the physical set-up of the table in the Vienna OMiLAB (were the *Scene2Model* approach was tested) is described. This description corresponds to the descriptoint, where the camera is poistioned underneath a transparent table top.

In the testing set-up the table is 97 cm high and the table top is 80cm x 80 cm. A picture of the table can be seen in figure 6.1. The camera is positioned 75,5 cm underneath the table. In the test implementation the *Logitech HD pro Webcam C920* USB camera was used for capturing the video stream. The front side of the paper figures

is showing in the direction of the x-axis.

For the tag recognition to work, also fixated tags on the table are necessary. In figure 6.1 these fix tags are marked with green circles. The markers must be put on the pre-defined position, if the standard configuration is used. If they are put on other positions, the new coordinates must be entered in the *Tag recognition software*. How it can be configured is described in section 4.3.3. The values for their x- and y-coordinates are always corresponding to the middle point of a tag. The following tags and their position must be used, if the standard configuration is taken. The values for the x- and y-axis are given in millimetres.

Positions of the fixed tags:

- id: 115, x:100 mm, y:100 mm
- id: 197, x:400 mm, y:100 mm
- id: 23, x:700 mm, y:100 mm
- id: 85, x:100 mm, y:400 mm
- id: 3, x:100 mm, y:700 mm
- id: 51, x:150 mm, y:150 mm
- id: 21, x:190 mm, y:190 mm

6.2 Paper figures

For using the automatic transformation also the paper figures must be prepared. Therefore, the figures must be cut out and adapted so that they can stand. Further, marker must be attached to them. For the stationary setting the tags must face downwards and for the mobile setting they must face upwards.

For the test implementation the paper figures from SAP Scenes^{TM} were used. For non-commercial purpose they can be used and adapted free. They are available under `https://experience.sap.com/designservices/approach/scenes` and licensed under the *Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License*, available under `https://creativecommons.org/licenses/by-nc-sa/4.0/` (Both internet addresses last visited on the 5-10-2018).

To better work with the paper figures, we put each figure in a group and made boxes in which the figures of a group are stored together.

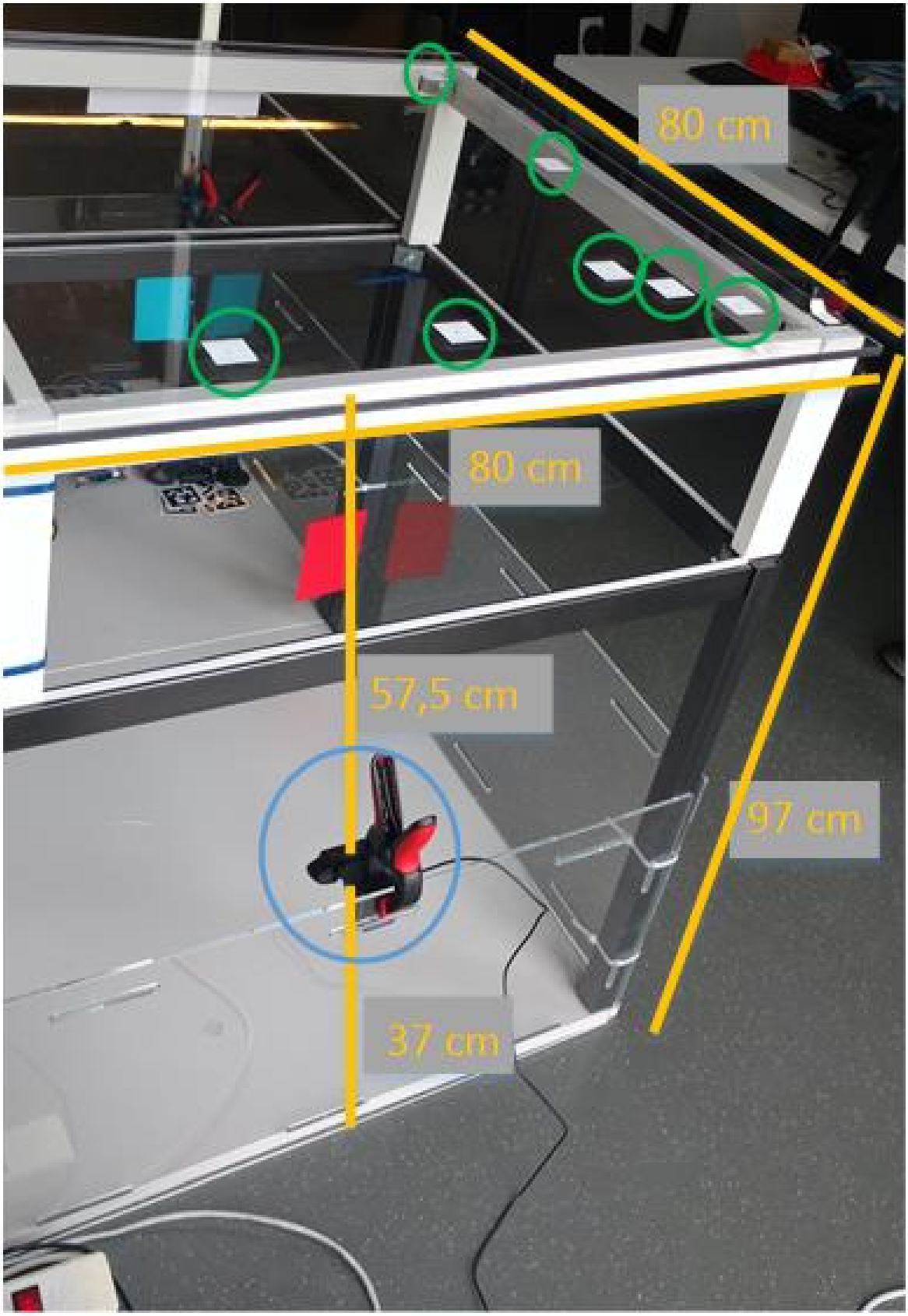

Figure 6.1: Picture of the table in test implementation of the *Scene2Model* approach

7 Contact

7.1 Technical Contact

Christian Muck (christian.muck@univie.ac.at)