VITRO – Model Based Vision Testing for Robustness

Oliver Zendel[1], Wolfgang Herzner[2], Markus Murschitz[3]
[1]AIT Austrian Institute of Technology GmbH, Austria, oliver.zendel@ait.ac.at
[2]AIT Austrian Institute of Technology GmbH, Austria, wolfgang.herzner@ait.ac.at
[3]AIT Austrian Institute of Technology GmbH, Austria, markus.murschitz@ait.ac.at
(Tel: +43-50550-4261, Fax: +43-50550-4150)

***Abstract*-- This paper introduces a model-based approach for testing the robustness of computer vision solutions with respect to a given task or application. The work is motivated by the observation that today testing computer vision components is mostly a manual and heuristic task. In general, recorded test images are used to assess whether the solution yields the correct output. This approach suffers from at least two deficiencies: (i) test data often need to be manually annotated with the expected output ("ground truth"), which is expensive and subjective; (ii) it is hard to assure that all situational aspects are included (e.g. backlight or occlusions) that can hamper the correct operation of the vision component. Our approach enables to generate test data with a measurable coverage of optical situations typical and critical for a given application, and also generates correct ground truth with no manual effort.**

***Index Terms*--computer vision testing, robustness testing, model-based test case generation, validation**

I. INTRODUCTION

Image processing techniques used in vision sensors are a crucial component for enabling autonomous systems to solve basic tasks of manipulating and navigating in an environment. In scenarios where the systems have to coexist with humans (e.g. robots taking care of the elder or autonomous cars in public traffic) all of their components should be certified to ensure that they operate safely. However, still today no general methodology exists for testing the robustness of vision components with a measurable coverage of scenes and situational aspects typical and critical for a given application. These coverage metrics are crucial to allow the certification of vision components. Public authorities require for the commercial application of safety-critical systems their safety certification. Due to the current lack of viable means to certify visual components or whole systems that depend on such components, the broad usage of autonomous systems in our everyday life is still highly restricted and hampered. It is thus clear that advances in methodology for the certification of vision components at this stage are as important to its development as improving the robustness of vision components and algorithms themselves.

When assessing the quality of a robot vision component (system under test; SUT) it is less an issue whether it produces the intended result; e.g. it is easy to check whether an edge detection algorithm actually outputs edges. It is also less complicated as conventional V&V techniques applicable to generic algorithms and systems can be used for the verification of vision algorithm implementation just as well (e.g. checking for the absence of faults that could lead to access violation, deadlock, or timeouts). The main question is how robust a solution is, i.e. how well it copes with the huge number of challenges present in the input data, e.g. shadows, reflections, occlusions, as well as artifacts emanating in the sensor, e.g. thermal noise – which we call *criticalities*.

Here the vision algorithm presents a very different challenge than the generic algorithm case. This is due to the immense number of possible different input images and the hard task of finding equivalence classes among images. For instance, with a given set of images it is hard to come up with an equivalence class that shows blue cars or no trees.

Today, in most cases recorded test images are used for validation from two kinds of sources:

On the one side, there exist a lot of publicly available data sets (e.g. the Brodatz texture album[1], the Middlebury data sets[2] or the KITTI Vision Benchmark Suite[3] of stereo images for disparity map testing, or the CAVIAR[4]and "Imageparsing.com" (IP)[5] data for indoor person tracking). In general, such test sets do not address a certain application. They are primarily used for benchmarking rather than for assessing application specific robustness. But even if they are application specific (e.g. the Color FERET Database, USA[6] for face recognition), they do not give an objective measure of the covered criticalities.

On the other side, test data is recorded especially for a given application. For instance, in the course of the ECV [7] project about 600.000 images were recorded for the testing of automatic plant and weed localization. For the quality assessment of welding robots [8], about 100.000 images were recorded. Despite their large number, a measurable coverage of criticalities is not available.

Finally, the expected output (or *ground truth*, GT) needs to be provided for the test data, in order to assess the SUTs response on the test input. This is usually done manually, which is expensive and at least subjective if not error prone. For instance, for the 60 test images used for comparing edge detectors in ([1]), several experts generated the GT manually, with clearly different results (see Fig. 1) and corresponding different assessments of the tested algorithms.

[1] http://www.ux.uis.no/~tranden/brodatz.html
[2] http://vision.middlebury.edu/stereo/
[3] http://www.cvlibs.net/datasets/kitti/
[4] http://homepages.inf.ed.ac.uk/rbf/CAVIAR/
[5] http://www.imageparsing.com/frame/MainShow1.html
[6] http://face.nist.gov/colorferet/

Fig. 1. Variation example in GT specified manually by two different persons. From ([1])

In order to overcome these deficiencies, we developed an approach for generating test data (both stimuli, i.e. images, and expected responses, i.e. GT) from models. This approach called VITRO (VIson Testing for Robustness) is justified by a number of additional observations:

- Computer graphics has reached a maturity status that allows to render realistic images;
- Criticalities can be included explicitly in generated data;
- Scenes can be created which would be too difficult or dangerous to be arranged in reality.

The resulting test data set should contain little redundancy and a measurable amount of domain as well as criticality coverage.

In the next section, this approach is outlined, while section II. presents VITRO, our new approach for model-based test case generation. Section III. contains some first preliminary results. Finally Section IV. contains a summary and some outlook.

II. The VITRO Approach

A. General Approach

The VITRO model-based test case generation process for computer vision is performed in these steps:

1. A *domain analysis* identifies objects that can appear in scenarios of the given application, together with their properties and relationships; all this information is specified in the so-called *domain model.*
2. *Criticalities* that should be included in the test data are either derived from the domain analysis, and/or selected from a *catalogue of criticalities*.
3. *3D-scenes* are automatically derived from the domain model such that both the *domain space* and the criticalities are covered.
4. *Images* are rendered from 3D-scenes and characterizations of the rendered images are computed.
5. The images are clustered according to the *characterization vector* and only the central *representatives* of each cluster are used as test data to minimize redundancy
6. GT and high quality renderings are generated for the representatives, *camera effects* are added by a post-processing step.

Figure Fig. 2 illustrates this process. The individual ingredients and steps are described in the following chapters.

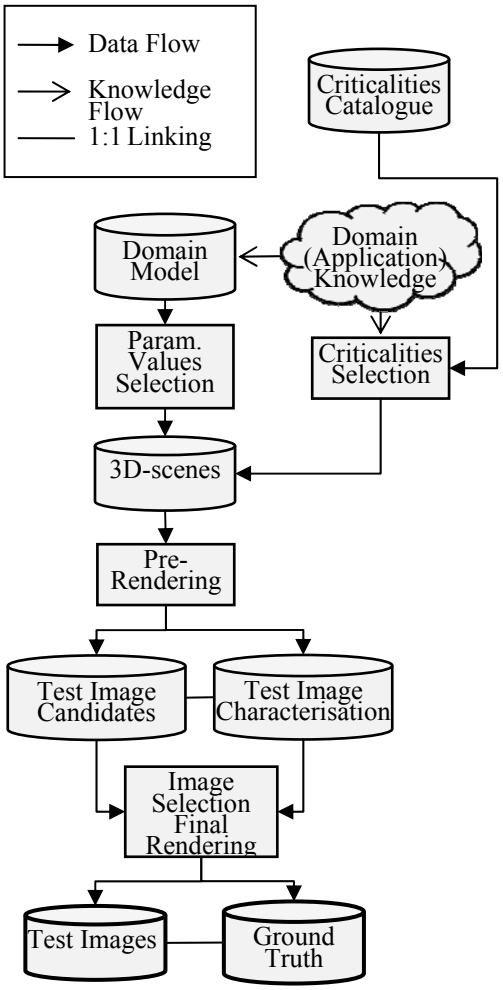

Fig. 2. Test case generation process

B. Domain Model

The domain model captures all aspects and information about the targeted application domain that are relevant for test image generation. The act of creating a specific domain model is called domain analysis. The domain model includes:

- Objects and their categories. For instance, objects and categories typical for an indoor application in home kitchens are: stoves, sideboards, tables, chairs, dishes, cups, pots, lamps, coffee machines, towels, doors, windows etc.
- Object properties specify shape and size, perhaps also variability-aspects (e.g. whether the object is flexible, and to what degree), color, texture, transparency, reflectance etc.

- Illumination. The light emitted from light sources has to be specified with respect to intensity, color, and distribution of light in different directions. Light sources, which can be seen by the SUT, have additionally to be modeled as objects.
- Media properties. If media such as air or water can influence the visual appearance of scenes, they have to be included in the domain model, characterized by properties such as dimming factor – e.g. fog, refraction index, or embedded particles – e.g. rain or snow.
- Relationships and constraints. Relationships describe how two or more objects are related to each other, e.g. "lid A belongs to pot B". Constraints place restrictions on objects and properties, e.g. that cups are placed with their bottom plane on horizontal surfaces, that chairs are aside the table, or that an analogue clock always has one hour hand and one minute hand, and optionally one additional second hand. A further aspect would be *behavior*, addressing motion and interactions. Behavior introduces the complex element of timing and conditional sequences, which are neglected during the course of this work, but will be addressed in future work.

The domain model is encoded as XML files in order to benefit from the huge amount of standardized tools. It is extensively using the xml-specifications of `xinclude`[7], `xpath` [8], and some `xslt` [9] transformations. `xinclude` supports model reuse and thus helps to reduce the high costs of domain model specification. Objects geometry and other properties, e.g. textures, are encoded using encodings well known in computer graphics, e.g. COLLADA[10]. Finally, so-called *object generators* are used to represent "generic objects" of a certain kind. Controlled by a rather small number of parameters, they allow generate a large variety of similar objects, e.g. cups or clouds.

C. Criticalities

In order to identify as many potential criticalities as possible, a so-called "CV HAZOP" has been carried out. For that purpose, a hazard analysis technique proven by the V&V community called HAZOP [5] was adapted by extending the set of usual guidewords with those that were considered appropriate (e.g. spatial (a)periodic) and applied to the generic vision algorithm. The "system" – including visible objects, light sources, media (e.g. air and its pollution), and the sensor (optics and recoding unit) – was modeled by an information-oriented approach. See [6] for a thorough description of this work including the complete resulting list of identified criticalities.

More than one thousand visual "criticalities" (i.e. situations and effects that can reduce the output quality of a CV algorithm) were identified and collected in a catalogue. Examples are "mirrors fake additional light sources" or "one object is split into multiple objects (fragments) because parts of the object are covered by a different object or are outside of the view". Experts from Siemens Munich, Fraunhofer-Gesellschaft Institute for Process Automation and AIT as well as others collaborated to create the CV HAZOP. Figure Fig. 3 shows some examples of identified criticalities.
During the domain analysis, those criticalities are identified that could be relevant for the given application. During this selection process, it is often feasible to also consider relevant characteristics of the CV solution to be tested.

Fig. 3. Example for typical criticalities: duplication due to mirroring, multiple shadows, confusing uniform texture

D. Parameter Values Selection and 3D-Scenes Generation

The domain model is represented by the so-called *domain space*, a high-dimensional parameter space where each of the domain parameters, e.g. object positions or intensity of light sources, establish its dimensions. A point in this space represents a very specific 3D-scene; if the parameters describing the observer (sensor) are included, it even specifies a certain view.
Theoretically, sampling the domain space with a sufficiently high density, yields test cases that contain all typical and critical situations for the given application. However, this does not answer when the sampling density is sufficient and also would generate an unknown fraction of redundant test data, thus violating the goal of creating an efficient test set. While even an outline of the whole sampling process as implemented is not possible here due to space limits, essential considerations are summarized below.

- To minimize the risk of missing relevant parameter value combinations due to too regular (e.g. on periodic raster positions) or too irregular (e.g. using Monte Carlo methods) sampling, sampling methods are used that provide low geometric discrepancy "lowD" ([4]).
- Since these methods work best in spaces of limited dimensionality, a somewhat hierarchical approach of splitting the domain space is used. First, the so-called "cardinality space" is lowD-sampled, which defines which and how many objects are added to a 3D-scene. This results in a parameter space specific for a family of 3D-scenes containing the same entities with

[7] http://www.w3.org/TR/xinclude/
[8] http://www.w3.org/TR/xpath/
[9] http://www.w3.org/TR/xslt/
[10] http://www.khronos.org/collada/

different values chosen for their parameters (e.g. location and rotation).

- This space is split into lower-dimensional subspaces that are as mutually independent as possible; e.g. parameters describing one object are collected in a common subspace. Each of these subspaces is lowD-sampled, and the resulting points combined under consideration of constraints (including physical laws such as that two solid bodies cannot intersect).

These considerations address "domain coverage", i.e. to find scenes typical for a given application. For "criticality coverage", further steps are carried out. One goal of the framework is to create test data that allows establishing characterization plots with respect to certain criticalities. For instance, by incrementally modifying the distance of an object or its occlusion fraction by another object, the sensibility of the SUT with respect to these aspects can objectively be assessed.

Another step, "criticality injection", considers that criticalities often concern interrelations between scene entities (objects, light sources, the observer etc.). Since the specified domain model is internally transformed into a Satisfiability Modulo Theory (SMT) expression ([3]), it is possible to generate solutions, i.e. 3D scenes, which contain the respective criticalities.

The decision when the sampling density is sufficient, i.e. when sampling can be stopped, is addressed in the next section. It should be noted that lowD-sampling algorithms are used that produce point sequences which are continuously lowD-distributed. Hence, sample point generation can be stopped at any time without violating the lowD property.

E. Parameter Rendering and Test Data Selection

The following approach is taken to avoid redundancy in the generated test data. In addition, this approach also provides a convergence criterion to detect when the test data generation process is finished (i.e. when we have generated "enough" test cases):

- Each individual 3D-scene is rendered using a high speed low-quality rendering engine. This step is synonymous to simulating a pin-hole camera that captures the respective 3D-scene. Figure Fig. 5 shows a wire-frame representation of a typical 3D model of a 3D-scene depicting a kitchen with a filled table as well as the low-quality rendering of the same scene.
- For each output image, a custom-build rendering system calculates an image characterization vector (also called *trait vector*) that can be used to describe the content of the output image in a very condensed and comparable form. All entries of the trait vector are normalized to the range [0;1]. This allows comparing two such vectors by calculating the distance between them (e.g. Euclidian distance) and thus allows to objectively measure similarities between different images based on the shown content. This characterization is possible due to the nature of having complete control and insight over the artificial generated input 3D-scene. Each pixel of the output image can be mapped unambiguously to the objects, lights or media that created it. Some examples for entries for the trait vector are: Proportion of the image that shows a specific object, proportion of surface of an object that is visible, orientation of an object that is visible in the current image, number or size of glare spots visible in the image, etc.
  The term trait vector was chosen to prevent its confusion with the term *feature vector*, a term used in CV to describe a description vector that describes a specific point within an image using pixel based metrics. An important difference here is that feature vectors have no background knowledge of the underlying objects that created those pixels, but can only work on the numerical pixel values themselves. The trait vector is filled with defined meta-knowledge stemming from the domain description thus making its entries correct by definition.
- The trait vectors of all candidates generated so far are clustered in the trait vector space with conventional clustering methods (see Figure Fig. 4). Convergence is assumed as soon as the adding of new candidates does not change the clustering over a predefined number of new candidates. Correspondingly a sufficient sampling density of the domain space is reached and no additional test cases have to be generated.
- From each cluster, a central representative is chosen (see Fig. 8 and Fig. 9. For these test cases, the individual test images are rendered using a high-quality rendering engine and also the necessary GT is generated (see Fig. 6).
  Figure Fig. 5 shows the difference in image realism of a low-quality and a high-quality rendering as well as a comparison to an actual real world camera image. The redundancy reduction step is also crucial to reduce the execution time of the test data generation. The low quality rendering of a scene, the calculation of the trait vector and the clustering can be done in less than a 10^{th} of a second while the high quality rendering can take up to an hour depending on the image size and the required level of realism. This huge speed factor explains why two different levels of realism are used in our approach.
- Nearly all rendering engines simulate a perfect pin-hole camera. A dedicated post-processing pipeline created by the Technical University of Brno is used to simulate real-world camera effects like depth of field and lens distortions (see [9] for a description).

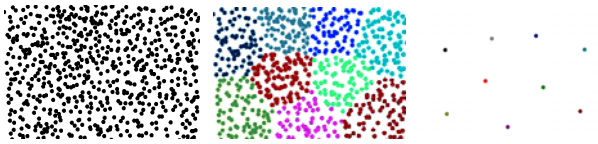

Fig. 4. Reducing redundancy by using central representatives of clusters[11]

Fig. 5. Comparison of the same scene: wire-frame representation, low-quality rendering, high-quality rendering and a genuine camera image

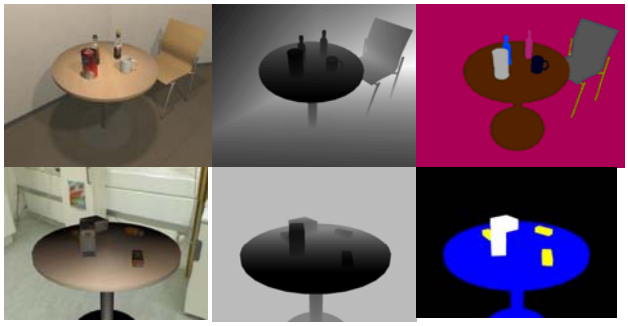

Fig. 6. Test data examples

III. Results

In first trail runs the VITRO framework was used to generate test cases for a use case defined in the R3-COP project. The task for a robotic platform is the detection and identification of different objects as well as their orientation that are placed on a kitchen table. This information is used to allow the robot to tidy up the table autonomously. In one test run, the different cardinalities and types of objects (e.g. cans and boxes) as well as different positions and orientations of the objects were tested. Figure Fig. 7. Effect of lowD-based domain space sampling (of cardinality and parameters of objects)Fig. 7 shows a selection from this data set. All mentioned parameters are sampled using a lowD-based approach as described in section II. D. creating a diverse and challenging data set. A more simple case was used for testing the clustering and convergence criteria described in section II. E. Here an identical 3D-scenery is sampled from various camera positions and camera orientations. A total of over 1000 different configurations were generated and the different clustering schemes and parameters were tested. Fig. 8 shows a selection from this dataset where borders of the same style indicate the correspondence to the same cluster with a clustering setting that enforced a total of nine clusters. Figure Fig. 8 shows the central representatives of each of these nine clusters.

[11] http://people.revoledu.com/kardi/tutorial/kMean/Image

Fig. 7. Effect of lowD-based domain space sampling (of cardinality and parameters of objects)

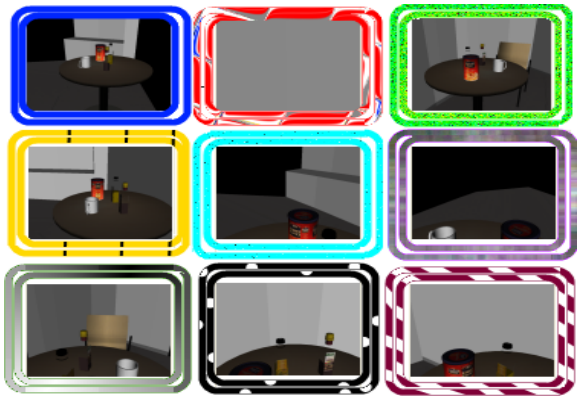

Fig. 8. Central representatives of each cluster shown in **Fehler! Verweisquelle konnte nicht gefunden werden.** using the same frame borders

IV. Conclusions

The presented approach VITRO for the model-based generation of test data for assessing the robustness of computer vision solutions with respect to a certain application has a number of strengths, challenges, and degrees of freedom.

Strengths:

- Difficult and dangerous scenes can be generated that would be very hard to arrange in reality.
- Typical and critical situations contained in test data can be measured.
- Ground truth can easily and objectively be generated.

Challenges:

- Creation or capturing of 3D-models of objects that are sufficiently precise for allowing realistic

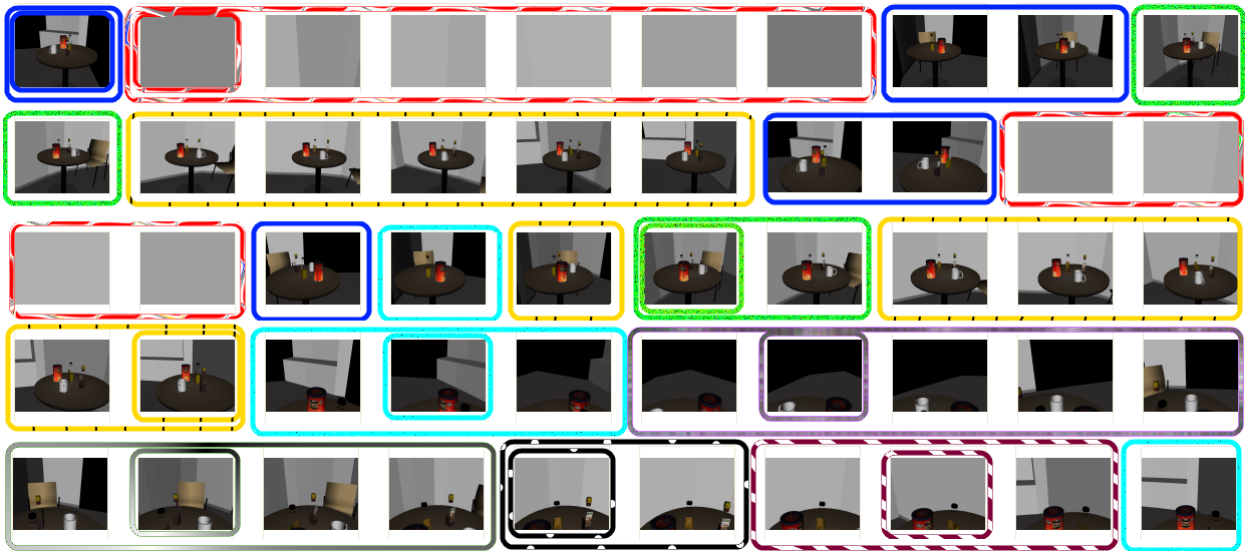

Fig. 9. Example of generated scenes, same borders indicate same cluster

rendering can be very expensive.

- Formalization of constraints and criticalities can become a complex task.
- Rendering should avoid rendering artifacts but include sensing artifacts (e.g. optical aberration, thermal noise); i.e. it shall be realistic for the SUT rather than for humans.

Degrees of freedom:

- Selection of criticalities.
- Selection of image characterization properties.
- Clustering in characterization vector space.
- Convergence criterion for clustering.

Further planned activities are:

- Gaining experience on the listed degrees of freedom.
- Support of SUT development. Test data could not only be generated for testing, but already for supporting its development by early provision of smart and relevant test data.
- Generating training data for learning algorithms. This needs to consider scene probabilities.
- Continuous test data generation. Provision of image sequences (videos) not only needs to include time, but may also require to consider behavior; possible even in closed-loop testing where the SUT is part of a sys-tem that, by reacting on the test data, changes the environment, which again has to be considered in the next test data generation step.
- Get the provided approach accepted for certifying computer vision solutions.

ACKNOWLEDGMENT

This work has been funded by the ARTEMIS project R3-COP (Resilient Reasoning Robotic Co-operating Systems), project number 100233.

REFERENCES

[1] Kevin Bowyer, Christine Kranenburg, and Sean Dougherty: “Edge Detector Evaluation Using Empirical ROC Curves”. In P. J. Flynn, A. Hoover, P. J. Phillips (eds.): Computer Vision and Image Understanding 84, 1–4 (2001); doi:10.1006/cviu.2001.0948; available online at http://www.idealibrary.com

[2] R. M. Haralick, K. Shanmugam, I. Dinstein: “Textural Features for Image Classification”. Systems, Man and Cybernetics, IEEE Transactions 3(6), 610–1621 (1973); doi: 10.1109/TSMC.1973.4309314

[3] Daniel Kroening, Ofer Strichman: “Decision Procedures – An Algorithmic Point of View”. Springer, Berlin Heidelberg, 2008. ISBN 978-3-540-741ß04-6

[4] Jiri Matousek: “Geometric Discrepancy – An Illustrated Guide”. Algorithms and Combinatorics Series, Vol.18; Springer, Berlin Heidelberg, 1999. ISBN 3-540-65528-X

[5] Felix Redmill, Morris Chudleigh, and James Catmur; “System Safety: HAZOP and Software HAZOP”. John Wiley & Sons, 1999; ISBN: 978-0-471-98280-7

[6] Oliver Zendel (ed.), Markus Murschitz (ed.): “Criticality Analysis for Computer Vision Algorithms – CV HAZOP”. R3-COP deliverable D4.2.3, 2012; available online at http://www.r3-cop.eu/ under “Download”.

[7] Croonen, Gerardus, and Csaba Beleznai: "Detection of near-regular object configurations by elastic graph search." Computer Vision and Graphics. Springer Berlin Heidelberg, 2010. 283-291.

[8] Fronthaler, H., Croonen, G., Biber, J., Heber, M., and Rüther, M.:„An online quality assessment framework for automated welding processes”. The International Journal of Advanced Manufacturing Technology, 1-10, 2013

[9] Michal Kučiš and Pavel Zemčík: "Simulation of Camera Features". Proceedings of CESCG 2012: The 16th Central European Seminar on Computer Graphics, Slovakia, 2012