Proceedings of the KI 2013 Workshop on Visual and Spatial Cognition

Marco Ragni, Michael Raschke, Frieder Stolzenburg (eds.)

SFB/TR 8 Report No. 034-08/2013
Report Series of the Transregional Collaborative Research Center SFB/TR 8 Spatial Cognition
Universität Bremen / Universität Freiburg

Contact Address:

Dr. Thomas Barkowsky
SFB/TR 8
Universität Bremen
P.O.Box 330 440
28334 Bremen, Germany

Tel +49-421-218-64233
Fax +49-421-218-64239
barkowsky@sfbtr8.uni-bremen.de
www.sfbtr8.uni-bremen.de

© 2013 SFB/TR 8 Spatial Cognition

Proceedings of the KI 2013 Workshop on Visual and Spatial Cognition

KIK – KI & Kognition Workshop Series

Marco Ragni, Michael Raschke, Frieder Stolzenburg (eds.)

Koblenz, 17-Sep-2013

Contents

Preface

The ability to process spatial information is crucial for various tasks as diverse as navigation, planning, and managing abstract concepts. Research issues in spatial cognition range from the investigation of human spatial cognition to mobile robot navigation. Much of that research effort has been experimental, putting little stress on precise models of the involved representations and processes. Obviously, spatial cognition is closely related to visual cognition of places and scenes in general. Different visualization techniques and reasoning formalisms serve to analyze spatial cognition processes. They can be used to achieve a more general cognitive model.

Since eye tracking devices became cheaper and their handling more comfortable during the last decade, eye tracking experiments are used in a wide field of user experiments for studying visual cognition. Many questions, however, remain open, for instance from an efficient analysis of recorded eye movements to the development of unified visual and cognitive models.

This workshop continues a series of successful workshops initiated by the Special Interest Group "Cognition" in the GI (German Society for Informatics). This sixth workshop, which is held in conjunction with KI 2013, aims at bringing together researchers from artificial intelligence, computer science, cognitive psychology, cognitive robotics, and visualization to foster a multidisciplinary exchange. The call was open for different topics and we received a variety of papers: from discussing the usage of low-level features for vision-based localization, to extracting spatial relations among objects for failure detection during plan execution of a robot, to challenges in using semantic knowledge for 3D object classification and semantic object recognition with segment faces. This interdisciplinary discussion of spatial and semantic models is enriched with contributions from eye tracking data visualization.

Our wish is that new inspirations and collaborations between the contributing disciplines will emerge from this workshop.

The organizers of this workshop would like to thank the organizers of the KI 2013 conference, especially Ute Schmid, and the Spatial Cognition Research Center SFB/TR 8 for their excellent support. We also would like to thank the members of the program committee for their help in selecting and improving the submitted papers, and finally all participants of the workshop for their contributions.

Marco Ragni, Michael Raschke, Frieder Stolzenburg

Organizers and Program Chairs

Marco Ragni, U Freiburg
Michael Raschke, U Stuttgart
Frieder Stolzenburg, HS Harz (contact person)

Program Committee / Reviewers

Thomas Barkowsky, U Bremen
Björn Browatzki, MPI für Kybernetik
Michael Burch, U Stuttgart
Lewis Chuang, MPI für Kybernetik
Christian Freksa, U Bremen
Reinhard Moratz, U Maine
Bernhard Nebel, U Freiburg
Thomas Röfer, DFKI Bremen
Ute Schmid, U Bamberg
Falk Schmidsberger, HS Harz
Stefan Wölfl, U Freiburg

KI 2013 Workshop on Visual and Spatial Cognition

Low-level global features for vision-based localization

Sven Eberhardt and Christoph Zetzsche

Cognitive Neuroinformatics,
Universität Bremen, Bibliothekstraße 1, 28359 Bremen, Germany
sven2@uni-bremen.de,zetzsche@informatik.uni-bremen.de

Abstract. Vision-based self-localization is the ability to derive one's own location from visual input only without knowledge of a previous position or idiothetic information. It is often assumed that the visual mechanisms and invariance properties used for object recognition will also be helpful for localization. Here we show that this is neither logically reasonable nor empirically supported. We argue that the desirable invariance and generalization properties differ substantially between the two tasks. Application of several biologically inspired algorithms to various test sets reveals that simple, globally pooled features outperform the complex vision models used for object recognition, if tested on localization. Such basic global image statistics should thus be considered as valuable priors for self-localization, both in vision research and robot applications.

Keywords: localization, visual features, spatial cognition

1 Introduction

The ability to make reliable assumptions about their own position in the world is of critical importance for biological as well as for man-made systems such as mobile robots. A number of sensors can be used and combined to achieve this feat (see e.g. [4]). Among these, vision is of particular importance. Although idiothetic information such as acceleration, velocity and orientation measurements can be used for dead reckoning, visual realignment can be essential to avoid the accumulation of errors in path integration. Furthermore, allothetic information in form of visual input can be used for direct localization. For example, many place cells in the hippocampus can be driven by visual input alone [10]. But how exactly can vision support localization?

The default hypothesis would be that this is achieved by just the same established principles of visual processing used for other spatial tasks like pattern discrimination or object recognition. The corresponding standard view of the visual system assumes that the main task of the system is invariant object recognition, and that this is achieved by a feed-forward system of feature extraction in form of a hierarchy of neural layers with increasing levels of abstraction and of spatial granularity [5, 6, 16]. This standard model is supported by numerous behavioral experiments and electroencephalography recordings, in particular by

experiments showing that human discrimination between categories in object and scene classification is achieved as early as 150ms after stimulus onset (for an overview see [16]).

In this paper, we look at vision-based self-localization from static allothetic input alone and formulate it as a classification problem: A set of example images per location is trained with their location as the label and the task is to attribute a new image to one of the learned locations by testing the classifier. Performance is evaluated by percent correct classified images, i.e. we disregard any metric information of distance between different locations and just treat each location as a class and all views from a location of instances in that class. From this perspective, the localization task is comparable to object classification problems such as the one posed by the Caltech-101 [3] dataset.

Generally, the features on which a classifier operates should be invariant to changes within a class but selective to changes between classes. Models designed for object recognition provide varying degrees of translation and scale invariance [13]. For example, the HMax features used for an animal detection task performed by [16] are designed to provide translation and scale invariance at local and global levels because animals may occur at different positions, sizes and 3D rotations in images.

However, whether object recognition and vision-based localization are really similar problems and can thus be solved with the same architecture has, to our knowledge, never been investigated systematically. In this paper, we ask whether visual features that are optimal for one task may be unsuited for the other and vice versa. To answer this question, we test how well feature outputs of a number of biologically inspired low-level vision models are able to discriminate among large numbers of locations and compare the results with benchmark performance on several object and scene recognition datasets.

2 Methods

Streetview dataset We use a novel dataset which has been sampled from Google Street View [1]. Street View has become popular as an outdoor dataset of natural scenes for self-localization, 3D map reconstruction, text recognition and image segmentation. Some unique key advantages to this dataset are its sheer amount of available data from many countries of the world, preprocessed in a standardized manner without bias to object centering [14]. Caveats include a bias to roads and populated areas as well as relatively poor image quality with distorted edges and Google watermarks.

204 locations are selected by picking random points in the sampling region until a road for which street view data is available is found within 50m range. For each location a full 360° yaw rotation in intervals of 10° for a total of 36 pictures per location is sampled. Field of view is 90° and pictures are stored as grayscale images with size 512x512 pixels. The Streetview dataset is sampled from random locations in France (SV-Country). To test localization on several distance scales, we generate two additional datasets from different sampling regions. For SV-City,

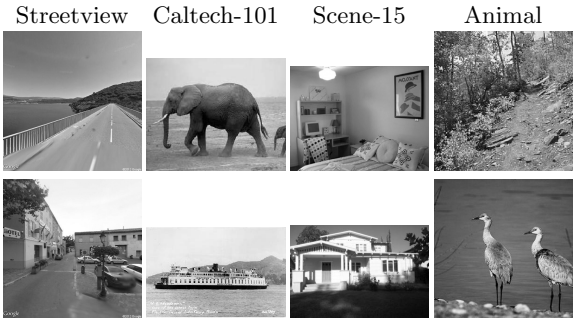

Fig. 1. Example images of the assayed datasets.

we sample locations in Berlin city center only. SV-World consists of imagery from all countries where street view was available.

Benchmark datasets To compare localization with object recognition and scene classification tasks, we also use several established categorization databases. The first dataset is Caltech-101 [3], which is a very diverse collection of 101 object categories containing between 31 and 800 images each. Categories are diverse and include specific animals, musical instruments, food categories, vehicles and more. Image contents vary between isolated objects, comic depictions and scenes containing the object in use. The dataset has been used as a benchmark for object recognition by a number of algorithms in the past, including an implementation of HMax and Spatial Pyramids. Caltech-101 is sometimes criticized because low-level algorithms can perform relatively well on some categories due to their very similar sample images [14]. However, the large number of categories alleviates this.

For a scene classification test, we use Scene-15 [7], which is a dataset comprised of photos of 15 different indoor and outdoor scene categories such as kitchen, forest and highway. Each photo shows an open scene without any objects close to the camera. Scene-15 has been mostly used to benchmark holistic feature extraction models such as Gist and Spatial Pyramids.

Finally, we include the Animal detection dataset from Serre et al. [16], which is a two-class classification object recognition dataset showing mostly non-urban outdoor scenes both with and without animals.

Models We focus on low-level, biologically inspired models that produce a fixed-size feature vector for each input image. For all models, we use implementation code supplied by the authors if available.

Textons by Malik et al. [9] apply a set of Gabor filters to an image, resulting in a response vector for each pixel. The response vectors are clustered into 128 textons and each pixel is assigned the cluster with the least square distance to its response vector. The resulting output vector is a histogram of these texton assignments

over the whole image. Textons have been used for image segmentation purposes [9] as well as scene classification [15].
Gist is also termed the *Spatial Envelope* of a scene by Oliva et al. It consists of the first few principal components of spectral components on a very coarse grid (8x8) as well as on the whole image. Gist has shown strong categorization performance on the Scene-15 dataset [12].
Spatial Pyramids, as described by Lazebnik et al. [7], calculate histograms over low-level features over image regions of different size and concatenates them to one large feature vector. The features used here are densely sampled SIFT [8] descriptors. For better comparability with the other models, we omit the custom histogram matching support vector machine (SVM) kernel used by Lazebnik in favor of a linear kernel and regression. We test the full pyramid up to level 2 (SPyr2) as well as outputs of the global histogram (SPyr0) only.
HMax is a biologically motivated multi-layer feed-forward model designed to mirror functionality found in the primate visual cortex ventral stream by Hubel and Wiesel [6]. It is based on the *Neocognitron* [5] and consists of alternating layers of simple and complex cells. Simple cell layers match a dictionary of visual patterns at all image locations and several scales, so units achieve selectivity to certain patterns. Complex cell layers combine the outputs of simple cells over a windows of locations and scales to achieve location and scale invariance. In this way, units of low layers have localized receptive fields and simple patterns, while units of higher layers respond to more complex patterns and are more translation and scale invariant. We use the CNS [11] implementation of HMax with parameter settings as chosen by Serre et al. [16]. The full feature vector of an image processed by HMax consists of randomly selected subsets of outputs of the C1, C2, C2b and C3 layers. In order to determine the effects of increasing invariance and matching to complex features, we also test performance when using only outputs of the C1, C2 and C3 layers respectively. To test if the task can be solved on trivial, low-level features, the classifier is also run on a luminance histogram and on a random subset of 2000 pixels from the images.
Classification is done on a normalized feature set which has been reduced to 128 features per image by principal component analysis. On these features, we perform regression with a linear kernel and leave-one-out cross validation to determine the regression parameter using the GURLS package [18] for MATLAB. Multi-class classification is performed by the one-versus-all rule. An equal number of training samples is taken at random from each class and all remaining elements are used for testing. Each run is repeated ten times with different test splits to yield the reported performance average and a standard deviation. Performance is defined as the percent correct averaged over all classes.

3 Results

All algorithms achieve between 28 and 76 percent correct performance on our dataset (Figure 2a). Performance ranges are similar to those found in the benchmark sets, which shows that our dataset has a comparable difficulty. Despite the dataset similarity in difficulty, we find that classification on Texton

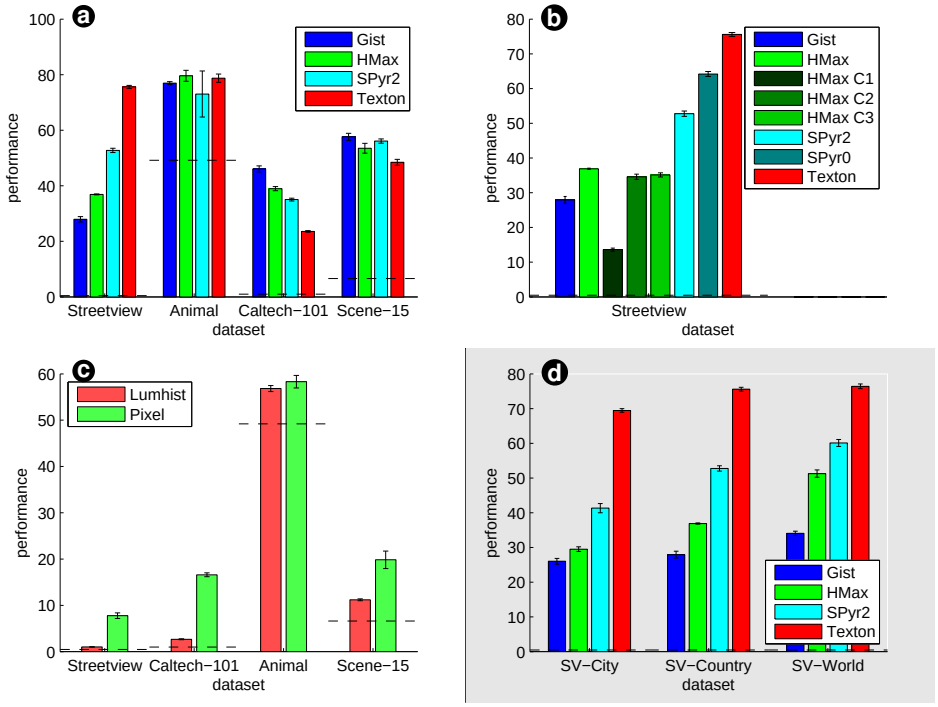

Fig. 2. Results performances of selected models and datasets in percent correct. Dashed lines mark chance level.

features yield the highest rank on all tasks of the streetview dataset, while they rank lowest on all other datasets. In particular, we do not observe this effect on the Scene-15 dataset, which hints that the requirements for scene classification are quite different from a true self-localization task. The strong performance of Textons is specially surprising, because they are the most basic and simple features in comparison with the outputs of HMax, Spatial Pyramids and Gist and they also output the least number of feature dimensions.

Spatial pyramids rank second on the performance scale. However, a test on the base level pyramid features (SPyr0 on Figure 2b) reveals that the performance at level zero of the pyramid exceeds that of the full pyramid at level two. Since the base level is just a histogram over densely sampled SIFT descriptors, classification actually happens based on a global histogram similar to that of the Textons. This means that any information about spatial arrangement of features is actually detrimental to self-localization performance.

The results suggest that the task is too easy in the sense that low-level features are sufficient to achieve high performance. However, tests on global luminance histograms as well as random image pixels show low performance near chance level (Figure 2c). In that sense, our self-localization dataset is harder than

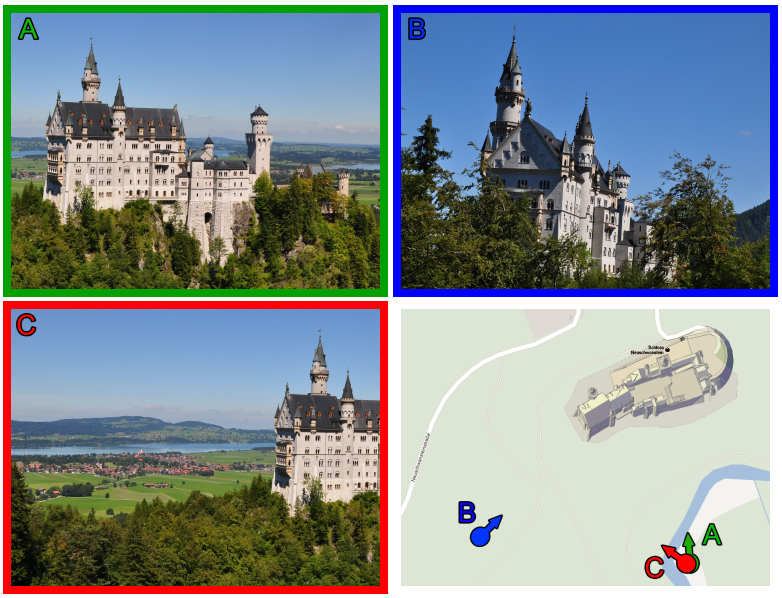

Fig. 3. Example for invariance requirements for object recognition versus localization: Views A and B show the same object from different locations, A and C show different views from the same location. An object classifier might pick up the similar castle features like towers and windows and put A and C into the same category. A self-localizer must not match such features and treat A and B as equal categories only.[1]

the benchmark datasets, for which 8-16% of all test samples could be classified based on raw pixel data alone.

Our findings generalize along different image sampling scales at SV-City, SV-Country and SV-World level (Figure 2d). Performance is higher at larger sampling scales, because locations are more different on a world scale than on a city scale. However, the performance order among different models remains the same.

4 Discussion

The results show quite clearly that model performance is highly task-dependent and there are no universal features that are optimal for any vision-based task. The main reason for this finding is that there are key differences in the invariance properties required for self-localization compared to those inherent to object or scene classification [2, 20].

While object recognition needs to be tolerant to changes in scale and rotation, self-localization does not (see Figure 3). Similarly, object recognition needs to be invariant to some feature rearrangements that occur when the object is seen

[1] Photos: ©Stephen & Claire Farnsworth via flickr, license CC-BY-NC. Map: Google maps ©Google inc.

from different angles. For self-localization, invariance to such rearrangements may be unwanted because if you see an object from a different angle, you are likely standing at a different position.

Concerning these invariances, HMax has both local translation and scale invariance built into the model. Thus it is not surprising that Streetview classification performance on these features is relatively poor. The differing invariance requirements also explain why neither Gist nor the pyramid structure of the spatial pyramid model could show strong performance on the dataset although both algorithms have been established for scene classification tasks [12]. Both models include features that are not completely location invariant, but contain the position in the image on a very coarse scale.

Classifying scenes in datasets like Scene-15 might actually be closer to a task like sorting photos, where photographers have a certain bias to how types of scenes are best portrayed and reflect that in the spatial arrangement of image features. Scene classification algorithms like Gist can catch on that common structure and use it for classification. However, when images from locations are recorded at random, unbiased angles, this method breaks down.

Although salient features are believed to be advantageous for localization [17, 19], we also find that the performance on complex SIFT descriptors is lower than on the more simple Textons. This is probably due to their high selectivity to particular objects, so they do not generalize well to matching on other, similar objects present in other views from the same location.

It appears surprising that Texton features, which have been designed for image segmentation [9], perform so well on a localization task. The reason seems to be that – among the models tested – they provide the best tradeoff between specificity to features present at individual locations and invariance to different views from the same location. The strong correlation of simple, global features with location suggests that very basic histogram features can be used as priors for self-localization algorithms for example in mobile robots instead of relying on geometric relations between complex features only. It also suggests that it might be worthwhile to check whether biological systems make use of such features to determine their own location.

Acknowledgments. This work was supported by DFG, SFB/TR8 Spatial Cognition, project A5-[ActionSpace].

References

1. Google Street View, `http://google.com/streetview`
2. Eberhardt, S., Kluth, T., Zetzsche, C., Schill, K.: From pattern recognition to place identification. In: Spatial cognition, international workshop on place-related knowledge acquisition research. pp. 39–44 (2012)
3. Fei-Fei, L., Fergus, R., Perona, P.: Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories. In: IEEE. CVPR 2004, workshop on generative model-based vision. vol. 106 (2004)
4. Filliat, D., Meyer, J.: Map-based navigation in mobile robots: A review of localization strategies. Cognitive systems research 4(4), 243–282 (2003)

5. Fukushima, K.: Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. Biological Cybernetics 36, 193–202 (1980)
6. Hubel, D., Wiesel, T.: Receptive fields and functional architecture of monkey striate cortex. The Journal of physiology pp. 215–243 (1968)
7. Lazebnik, S., Schmid, C., Ponce, J.: Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In: Computer vision and pattern recognition, 2006 IEEE computer society conference on. vol. 2, pp. 2169–2178. Ieee (2006)
8. Lowe, D.: Object recognition from local scale-invariant features. In: Computer vision, 1999. The proceedings of the seventh IEEE international conference on. vol. 2, pp. 1150–1157 (1999)
9. Malik, J., Belongie, S., Leung, T., Shi, J.: Contour and texture analysis for image segmentation. International journal of computer vision 43(1), 7–27 (2001)
10. Markus, E.J., Barnes, C.a., McNaughton, B.L., Gladden, V.L., Skaggs, W.E.: Spatial information content and reliability of hippocampal CA1 neurons: effects of visual input. Hippocampus 4(4), 410–421 (1994)
11. Mutch, J., Knoblich, U., Poggio, T.: CNS: a GPU-based framework for simulating cortically-organized networks. Tech. Rep. MIT-CSAIL-TR-2010-013 / CBCL-286, Massachusetts Institute of Technology, Cambridge, MA (2010)
12. Oliva, A., Hospital, W., Ave, L.: Modeling the shape of the scene: A holistic representation of the spatial envelope. International journal of computer vision 42(3), 145–175 (2001)
13. Pinto, N., Barhomi, Y., Cox, D.D., Dicarlo, J.J.: Comparing state-of-the-art visual features on invariant object recognition tasks. In: Applications of computer vision (WACV), 2011 IEEE workshop on. pp. 463–470 (2011)
14. Ponce, J., Berg, T.L., Everingham, M., Forsyth, D.A., Hebert, M., Lazebnik, S., Marszalek, M., Schmid, C., Russell, B.C., Torralba, A., Williams, C.K.I., Zhang, J., Zisserman, A.: Dataset issues in object recognition. Springer Berlin Heidelberg (2006)
15. Renninger, L.W., Malik, J.: When is scene identification just texture recognition? Vision research 44(19), 2301–2311 (2004)
16. Serre, T., Oliva, A., Poggio, T.: A feedforward architecture accounts for rapid categorization. Proceedings of the national academy of sciences 104(15), 6424–6429 (2007)
17. Sim, R., Elinas, P., Griffin, M., Little, J.: Vision-based SLAM using the Rao-Blackwellised particle filter. In: IJCAI workshop on reasoning. pp. 9–16 (2005)
18. Tacchetti, A., Mallapragada, P.K., Santoro, M., Rosasco, L.: GURLS: a toolbox for large scale multiclass learning. In: Big learning workshop at NIPS (2011), `http://cbcl.mit.edu/gurls/`
19. Warren, D.H., Rossano, M.J., Wear, T.D.: Perception of map-environment correspondence: The roles of features and alignment. Ecological psychology 2(February 2013), 131–150 (1990)
20. Wolter, J., Reineking, T., Zetzsche, C., Schill, K.: From visual perception to place. Cognitive processing 10, 351–354 (2009)

KI 2013 Workshop on Visual and Spatial Cognition

Extracting Spatial Relations Among Objects for Failure Detection

Mustafa Ersen[1], Sanem Sariel-Talay[1], and Hulya Yalcin[2]

[1] Artificial Intelligence and Robotics Laboratory, Computer Engineering Department
[2] Electronics and Communication Engineering Department
Istanbul Technical University, Turkey
{ersenm,sariel,hulyayalcin}@itu.edu.tr

Abstract. A cognitive robot may face failures during the execution of its actions. These failures are mostly due to the gap between the physical world and the constructed symbolic plans, some internal problems that may occur in its embodiment or unexpected external events. In this paper, we propose a visual scene interpretation system for extracting spatial relations among objects in a scene and using these relations to detect failures during the plan execution. Our system uses LINE-MOD and HS histograms in order to recognize textureless objects with different shapes and colors. Then, it analyzes the scene to specify the world state after each action execution. Our focus in this research is on particularly the following spatial relations: *on*, *on_table*, *clear* and *unstable*. In the experiments, we test the performance of our system on recognizing objects, determining pairwise spatial relations among them, and detecting failures using these relations. Our preliminary results reveal that our system can be successfully used to extract spatial relations in a scene, and to determine failures during plan execution by using this information.

Keywords: cognitive robots, failure detection, spatial reasoning, object recognition, automated planning

1 Introduction

A cognitive robot possesses abilities to construct symbolic plans to solve given problems and to execute these plans in the real world. Automated planners are commonly used for determining a valid sequence of actions for a robot to achieve its goals. These planners use high-level description of the problem and the domain (i.e. initial/goal states and operators corresponding to real-world actions) to construct a plan. After obtaining a valid plan, the robot needs to execute the corresponding real-world actions in order to attain the desired goal. However, it may face several types of failures during the execution of its actions in the real world [1]. These failures may arise due to the gap between the real-world facts and their symbolic representations used during planning, unexpected events that may change the current state of the world or internal problems.

Ensuring robostness is crucial for a cognitive robot in order to accomplish the given goals in the real world. In this work, we investigate how spatial relations

among objects are determined using visual data from an RGB-D camera and how this information is used to detect action execution failures in the real world. As a motivating example to illustrate the stated problem, consider the object manipulation task in the blocks world domain. An example plan constructed for a 3-block problem is given in Figure 1. In this toy problem, the aim is stacking three blocks on top of each other where all blocks are initially on the table and without any other objects on top of them (i.e. satisfying *clear* predicate). During the execution of the generated plan, the robot may fail in executing action *stack*. Possible reasons for this failure might be the weight of the object, improper grasp position or a vision problem. To ensure robustness in such cases, the robot needs to continuously monitor the state space for anomalies during action execution.

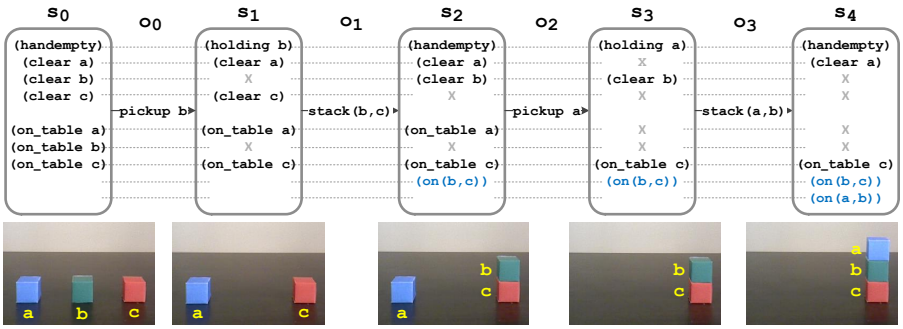

Fig. 1. The execution trace for solving the blocks world problem with a three block case is given. (top) The successor states and the actions taken at each state, (bottom) the visual scene observed at each world state are presented.

Throughout the paper, we first give some background information on the areas of automated planning, object recognition and scene interpretation. Then, we describe the details of our system for determining spatial relations among the objects in order to detect failures. We then give empirical results of our approach followed by the conclusions.

2 Background

In this section, we formulate the planning problem and give a brief review of the approaches used for recognizing objects and determining spatial relations in the scene. Then, in the following section, we present our solution to the stated problem.

2.1 Automated Planning

Cognitive robots may use automated onboard action planning for online generation of action sequences to accomplish given tasks against exogenous events.

A planning task Π can be described on a state space S containing a finite and discrete set of states including an initial state s_0 and a goal state s_G, and a state transition function $s_{t+1} = f(o_t, s_t)$ where $o_t \in O(s_t)$ is an operator applied in state s_t. State transition function is realized through planning operators $o \in O$ that are defined as symbolically abstracted representations of real world actions $a \in A$. A planning operator can be formalized as a tuple $o = \{pre(o), add(o), del(o), cost(o)\}$ where $pre(o)$ defines the preconditions, $add(o)$ and $del(o)$ define the effects of the operator and $cost(o)$ represents the cost of the corresponding action. $o_t \in O(s_t)$ is defined as the set of applicable operators in a state $s_t \in S$ determined by checking preconditions of the operators to satisfy $pre(o_t) \subseteq s_t$. By applying o_t at state s_t, a new state $s_{t+1} = add(o_t) \cup (s_t \setminus del(o_t))$ is observed. Planning task is achieved by a planner to reach s_G from s_0 by selecting a sequence of operators from $O(s_t)$ at successive states s_t and executing the corresponding actions $a_t \in A$ in the given order. After searching the whole space of operators, the planner constructs a valid plan $P = o_{0:G}$ by considering an optimization criteria (e.g., makespan) and the duration/cost of each operator. Having generated a valid plan P, the robot can execute each corresponding action $o_t \rightarrow a_t \in A$ in sequence in the physical world. If all goes well with execution, the robot successfully attains s_G. However, due to non-deterministic actions and different sources of uncertainty in physical environments, several failures may be encountered [1]. Our primary focus is action execution failures. To detect a failure, the robot should monitor its execution, recognize the objects it interacts with (if any) and interpret the scene continuously.

2.2 Object Recognition

There are various approaches for recognition of objects in a scene using different types of visual clues. These approaches can be categorized as 2D object recognition approaches based on local invariant feature descriptors and 3D object recognition approaches based on surface normals computed from the depth map. In the case of 2D color data, local feature descriptors are used to determine patterns in the image which differ from the other pixels in their neighborhood. These distinguishing parts of the image (i.e. keypoints) are generally chosen by considering sharp changes in color intensity and texture. To store the keypoints, descriptors are computed around them which are suitable for measuring their similarity. The idea of using local invariant descriptors became popular when Scale-Invariant Feature Transform (SIFT) [2] was proposed in 1999. SIFT is a keystone in the area, and it is used as the base of the state of the art techniques. It is known to be invariant against geometric transformations such as scale, rotation, translation and affine transformation to a sufficient extent for a lot of applications. It is also claimed to perform well against noises and changes in the illumination. However, SIFT-based approaches are known to have deficiencies in recognizing textureless objects. Information on the 3D shapes of the objects and their colors can be used in order to deal with this problem. By the development of RGB-D sensors, it is possible to get depth information as well as color

and texture information for this purpose. To utilize the depth values captured using these types of sensors, different 3D descriptors have been proposed [3]. These descriptors can be divided in two categories: local descriptors and global descriptors. Local descriptors are used to describe the local geometric properties of distinguishing points (i.e., keypoints) whereas global descriptors capture depth-based features globally for a presegmented object without storing local information for extracted descriptors. Among these, LINE-MOD [4] is unique as it is a linearized multi-modal template matching approach based on weak orientational features which can be used to recognize objects very fast making this approach the most suitable one for real-time robotic applications.

2.3 Determining Spatial Relations

Detecting and representing structures with spatial relations in a scene is known as the scene interpretation problem. While this is a trivial task for humans, interpreting spatial relations by processing visual information from artificial vision systems is not a totally solved problem for autonomous agents [5]. In the recent years, some approaches have been proposed to solve this problem [6–9]. Some of these works use 2D visual information for extracting qualitative spatial representations in a scene [6, 7]. In these works, some topological and orientational relations among objects are determined in the scene. In another work, proximity-based high-level relations (e.g., relative object positions to find objects that are generally placed together) are determined by comparing Euclidean distance between pairs of recognized objects in the scene [8]. This system relies on 3D data obtained using an RGB-D sensor and an ARToolkit marker acting as a reference coordinate system. Sjöö et al. have proposed a method for determining topological spatial relations *on* and *in* among the objects, and this information is used to guide the visual search of a robot for the objects in the scene [9]. Object recognition approach used in their work is based on matching SIFT [2] keypoints on a monocular image of the environment.

Our proposed work differs from the previous studies in two ways. First, determining spatial relations is done for a higher level task of detecting failures after action executions. Second, the object recognition system used in this work is more generic as it can deal with textureless objects that do not have any distinguishing texture information.

3 Scene Interpretation for Monitoring Action Executions

We propose a failure detection system based on visual information. The system involves three main procedures, namely, object recognition, scene interpretation and failure detection. In the system, first, each object of interest is modelled by creating multi-modal LINE-MOD [4] templates from different viewpoints. A template involves the surface normals within an object and the color gradients around its borders. As well as the multi-modal templates of LINE-MOD, a color histogram is generated to model each template in Hue-Saturation-Value

(HSV) color space. In this histogram, V(value) is omitted as it is strongly dependent on illumination conditions, and normalized values are taken for H(hue) and S(saturation). By using these histograms, object recognition process is improved as color values inside the templates are also considered. Then, LINE-MOD templates for all the objects of interest are searched in the scene using a sliding window approach to find matches. The threshold is specified as 80% by taking into account the noisy data captured using an RGB-D sensor. These matches are then verified comparing HS histograms of corresponding templates with match regions based on normalized correlation, and false positives are eliminated. The threshold value is taken as 0.5 by considering the changes in the illumination.

After the objects are recognized and located in the scene, qualitative spatial relations are determined for failure detection. In the blocks world domain, these relations are *on*, *on_table*, *clear* and *unstable*. Initially all the recognized objects are assumed to be *on_table* and *clear*. Then the *on* relation is determined between each pair of objects as follows,

$$\forall obj_i, obj_j, (EC(obj_i, obj_j) \vee PO(obj_i, obj_j)) \wedge N(obj_i, obj_j) \Rightarrow on(obj_i, obj_j)$$

where *EC(externally connected)* and *PO(partially overlapping)* are topological predicates of RCC8 [10] and *N(north)* is a directional predicate of cardinal direction calculus [11]. After determining the *on* relation, *clear* and *on_table* relations are updated for the objects involving in this relation as follows,

$$\forall obj_i, obj_j, on(obj_i, obj_j) \Rightarrow \neg on\_table(obj_i) \wedge \neg clear(obj_j)$$

To eliminate false positives in the extraction of *on_table* relations due to recognition failures, the area under the object is checked in order to see if it is planar or not. If the area is not a horizontal plane, than it is assumed that the corresponding object is not on the table. Finally, *unstable* relation is determined by checking the horizontal projections of the aligned object templates. If the horizontal projection of the upper object in the *on* relation has an unsupported part (i.e., out of the area covered by the object below) of more than 1/4 ratio to its length, this *on* relation is assumed to be *unstable*.

Action execution failures are detected by checking the state of the world with respect to spatial relations after each action execution. We consider three states after executing an action [12], namely *success*, *fail-safe* and *fail-unsafe* of which we repeat definitions here for convenience.

Definition 1 (*success* state) If all the desired effects of the action occurs in the environment, the situation is specified as *success*.

Definition 2 (*fail-safe* state) If the state of an execution is not *success* but the state does not change, the situation is specified as *fail-safe*. For example, the robot fails in picking up an object but the state of the object is not changed.

Definition 3 (*fail-unsafe* state) If the execution of an action fails and there is any damage and/or dangerous situation (e.g., an undesirable state is observed) or the robot cannot judge whether there is any harmful situation, the situation is specified as *fail-unsafe*. For example, the robot fails in picking up an object, and the object is broken into pieces or fallen down the ground out of reach of the robot.

4 Experimental Evaluation

In the experiments, the proposed system is evaluated in real time for different possible situations in the scene using the real-world data captured by an RGB-D sensor. The objects used in these experiments are three paper blocks, two plastic toy grapes and a single toy box (Figure 2). These objects are selected as they have different shape and/or color features.

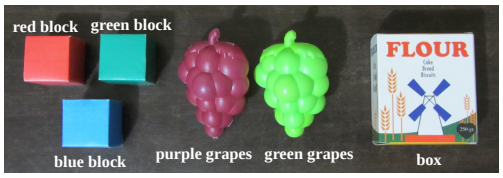

Fig. 2. The objects used in the experiments.

First, the overall recognition performance has been evaluated by comparing the results of LINE-MOD and our approach combining LINE-MOD with HS histograms. The results are illustrated in Table 1 as a confusion matrix for 120 different scenes (20 scenes for each object). As expected, both LINE-MOD and our approach give good recognition rate for different shaped objects. However, LINE-MOD cannot always distinguish similar shaped objects with different colors. Our approach based on checking HS histogram correlations on the results obtained using LINE-MOD leads to much better results in these situations. False negatives in recognition are slightly greater in our approach since some correct results are eliminated by checking color correlation.

Table 1. Confusion matrix for recognition: LINE-MOD / LINE-MOD&HS histograms.

	red block	*green block*	*blue block*	*green grapes*	*purple grapes*	*box*	*not found*
red block	12/19	3/0	4/0	0/0	0/0	0/0	1/1
green block	2/0	13/18	4/0	0/0	0/0	0/0	1/2
blue block	5/0	2/0	12/18	0/0	0/0	0/0	1/2
green grapes	0/0	0/0	0/0	13/20	7/0	0/0	0/0
purple grapes	0/0	0/0	0/0	6/0	13/18	0/0	1/2
box	1/0	2/0	0/0	0/0	0/0	17/20	0/0

Second, the performance of our system for extracting spatial relations: *on*, *on_table*, *clear* and *unstable* has been tested in an experiment involving 100 scenes (50 scenes for the blocks, 50 scenes for the grapes and the box). The results are shown in Figure 3. As given in these results, our system can be used to successfully detect relations for all the objects used in our tabletop scenarios. The highest error is in determining *on* relation by 20% and this is caused by the objects that cannot be recognized. When the object that is located below another object cannot be recognized, this also affects the success of determining *on_table*

relation. Similarly, there are some errors in determining *clear* relation for an object when another object that is located on top of it cannot be recognized. Errors in *unstable* relation are due to the failures in recognition or bad alignment of the recognized templates.

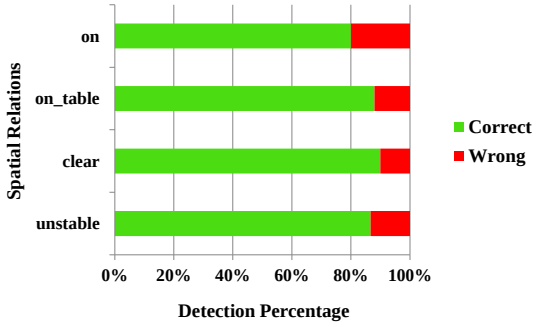

Fig. 3. Performance of the proposed system on determining spatial relations.

In the last set of experiments, the performance of the failure detection process has been tested on 20 different scenes that are set for each execution state: *success*, *fail-safe* and *fail-unsafe*. Similar to the previous set of experiments, it has been observed that, spatial relations are determined correctly in 85% of the scenes where the action is executed successfully. Moreover, the system has been observed to label 90% of the scenes with a *fail-safe* state correctly where the state of the world does not change after executing an action. In these experiments, when the object to be stacked is fallen down the table and this object cannot be recognized in the scene, the state is assumed to be a *fail-unsafe* state. With this assumption, the system has identified all *fail-unsafe* examples in the 3-blocks problem correctly.

5 Conclusion

We have presented an approach for detecting failures to ensure robust task execution in cognitive robotic applications. Our approach is based on using visual information extracted from the scene in order to determine spatial relations among the objects that are involved in manipulation scenarios. First, we have shown how our system can be used to recognize objects with different geometric shapes and colors. Then, we have given the details of the visual scene interpreter for specifying spatial relations among the objects of interest and evaluating these relations for detecting failures during action executions. The preliminary results of the conducted experiments on our system indicate that the system can be used to successfully detect states with failures in an object manipulation scenario. In

our future studies, we plan to conduct experiments on larger sets of scenes involving various objects to justify our research. Our ongoing work includes the integration of temporal reasoning into spatial reasoning in order to detect the possible causes of failures from previous states (e.g., an unstable stack of blocks causing a failure when stacking another block on top of them).

6 Acknowledgement

This research is funded by a grant from the Scientific and Technological Research Council of Turkey (TUBITAK), Grant No. 111E-286. We thank Prof. Muhittin Gokmen for his recommendations on vision algorithms.

References

1. Karapinar, S., Altan, D., Sariel-Talay, S.: A robust planning framework for cognitive robots. In: Proc. of the AAAI-12 Workshop on Cognitive Robotics. (2012) 102–108
2. Lowe, D.G.: Object recognition from local scale-invariant features. In: Proc. of the 7th IEEE Intl. Conference on Computer Vision (ICCV'99). (1999) 1150–1157
3. Aldoma, A., Marton, Z.C., Tombari, F., Wohlkinger, W., Potthast, C., Zeisl, B., Rusu, R.B., Gedikli, S., Vincze, M.: Tutorial: Point cloud library: Three-dimensional object recognition and 6 DOF pose estimation. IEEE Robotics and Automation Magazine **19**(3) (2012) 80–91
4. Hinterstoisser, S., Cagniart, C., Ilic, S., Sturm, P.F., Navab, N., Fua, P., Lepetit, V.: Gradient response maps for real-time detection of textureless objects. IEEE Trans. on Pattern Analysis and Machine Intelligence **34**(5) (2012) 876–888
5. Neumann, B., Möller, R.: On scene interpretation with description logics. Image and Vision Computing (Cognitive Vision Special Issue) **26**(1) (2008) 82–101
6. Falomir, Z., Jiménez-Ruiz, E., Escrig, M.T., Museros, L.: Describing images using qualitative models and description logics. Spatial Cognition and Computation **11**(1) (2011) 45–74
7. Sokeh, H.S., Gould, S., Renz, J.: Efficient extraction and representation of spatial information from video data. In: Proc. of the 23rd Intl. Joint Conference on Artificial Intelligence (IJCAI'13). (2013) 1076–1082
8. Kasper, A., Jäkel, R., Dillmann, R.: Using spatial relations of objects in real world scenes for scene structuring and scene understanding. In: Proc. of the 15th IEEE Intl. Conference on Advanced Robotics (ICAR'11). (2011) 421–426
9. Sjöö, K., Aydemir, A., Jensfelt, P.: Topological spatial relations for active visual search. Robotics and Autonomous Systems **60**(9) (2012) 1093–1107
10. Randell, D.A., Cui, Z., Cohn, A.G.: A spatial logic based on regions and connection. In: Proc. of the 3rd Intl. Conference on Principles of Knowledge Representation and Reasoning (KR'92). (1992) 165–176
11. Frank, A.U.: Qualitative spatial reasoning with cardinal directions. In: Proceedings of the 7th Austrian Conference on Artificial Intelligence. (1991) 157–167
12. Karapinar, S., Sariel-Talay, S., Yildiz, P., Ersen, M.: Learning guided planning for robust task execution in cognitive robotics. In: Proc. of the AAAI-13 Workshop on Intelligent Robotic Systems. (2013)

KI 2013 Workshop on Visual and Spatial Cognition

Semantic Object Recognition with Segment Faces

Falk Schmidsberger and Frieder Stolzenburg

Harz University of Applied Sciences, Automation and Computer Sciences Department, Friedrichstr. 57-59, 38855 Wernigerode, Germany
{fschmidsberger,fstolzenburg}@hs-harz.de

Abstract. Recognizing objects from images becomes a more and more important research and application topic. There are diverse applications such as face recognition, analysis of aerial images from multicopters, object tracking, image-based web search, etc. Many existing approaches focus on shape retrieval from a single polygon of contour points, or they try to compare clouds of interesting points of an object. However, human object recognition concentrates on few points of the segments forming the object. Clearly, complex objects, strictly speaking the projections of their shape on the image plain, consist of several (polygonal) segments. Therefore, the procedure presented in this paper takes this into account, by composing objects hierarchically into a group of segments. We briefly introduce our procedure for semantic object recognition based on clusters of image segment contours and discuss the problem of recognizing objects from different perspectives.

1 Introduction

Already in [2], it has been stated, that the contour points of an object, for instance of a cat, are of particular importance for human semantic object recognition. By semantic, we mean in this context that we do not only want to recognize abstract geometric forms but real complex objects, given by (non-preprocessed) example images, which are not characterized by a single contour, but by a group thereof. The surface of a complex object can be considered as consisting of several polygons. When such an object is viewed from a certain viewpoint, its image may be perspectively distorted. Nevertheless, several features remain invariant, for instance segment neighborhood relations, among others. Therefore, we focus in our approach on segment contours and their adjacency relations. Our overall procedure of object recognition roughly works as follows (cf. [13]):

Each object in an image is decomposed into segments with different shapes and colors. In order to recognize an object, e.g. a house, it is necessary to find out which segments are typical for this object and in which neighborhood of other segments they occur. A group of typical and adjacent segments for a certain object defines the whole object in the image. Similar segments are clustered. A hierarchical composition of these segment clusters enables model building, taking into account the spatial relations of the segments in the image. The

procedure employs methods from machine learning, namely k-means clustering and decision trees with boosting [3, 5, 9], and from computer vision, e.g. image pyramid segmentation and contour signatures [4].

The rest of this paper is organized as follows: First, we introduce our procedure for semantic object recognition in some more detail (Sect. 2). After that, we consider the influence of perspective projection on object recognition (Sect. 3). Next, we provide a brief evaluation of the approach (Sect. 4) and discuss some other approaches on object recognition (Sect. 5). Finally, we summarize and conclude the paper (Sect. 6).

2 Semantic Object Recognition

In our procedure for semantic object recognition, at first, we train models with sample images for each object category. After training, the models are used to recognize objects in other images. This is done as sketched in Fig. 1. Most of the steps and data in this process (marked in black in the figure) are identical for the training and the recognition phase. The steps and data that are relevant only for the training phase are marked with blue boxes and arrows, whereas the data and steps that are relevant only for the recognition phase are marked in green.

1. **Image optimization and segmentation:**
   For each pixel in the image, similar neighboring pixels are colored with a uniform color by a flood fill algorithm. With an image pyramid segmentation algorithm, the shapes of the resulting blobs of uniform color are extracted as image segments [4].
2. **Segment feature vector extraction and normalization:**
   A feature vector is computed for each segment, using the data of four normalized distance histograms, computed from the segment contour. A distance histogram consists of a vector of distances computed with several related methods: polar distance, contour signature, and ray distance [1, 10, 15] (see Fig. 2).
3. **Compute/use cluster models over the feature vectors:**
   During *training*, a cluster model with all feature vectors from all images of one category is created. Each cluster represents a familiar segment of the actual object category. During *recognition*, the cluster model of a category is used to select the familiar segments in the actual image.
4. **Segment tree creation, using only familiar segments of a category:**
   A segment tree comprises typical adjacent segments and the hierarchical composition of segment clusters for a certain object. It defines the whole object in the image. This histogram method is invariant against translation, rotation, scaling, and partially also to perspective distortions.
5. **Compute/use the decision tree models:**
   During *training*, a decision tree model from the data of the segment tree for each object category is created. During *recognition*, the decision tree model of each category is used to recognize objects in the actual image.

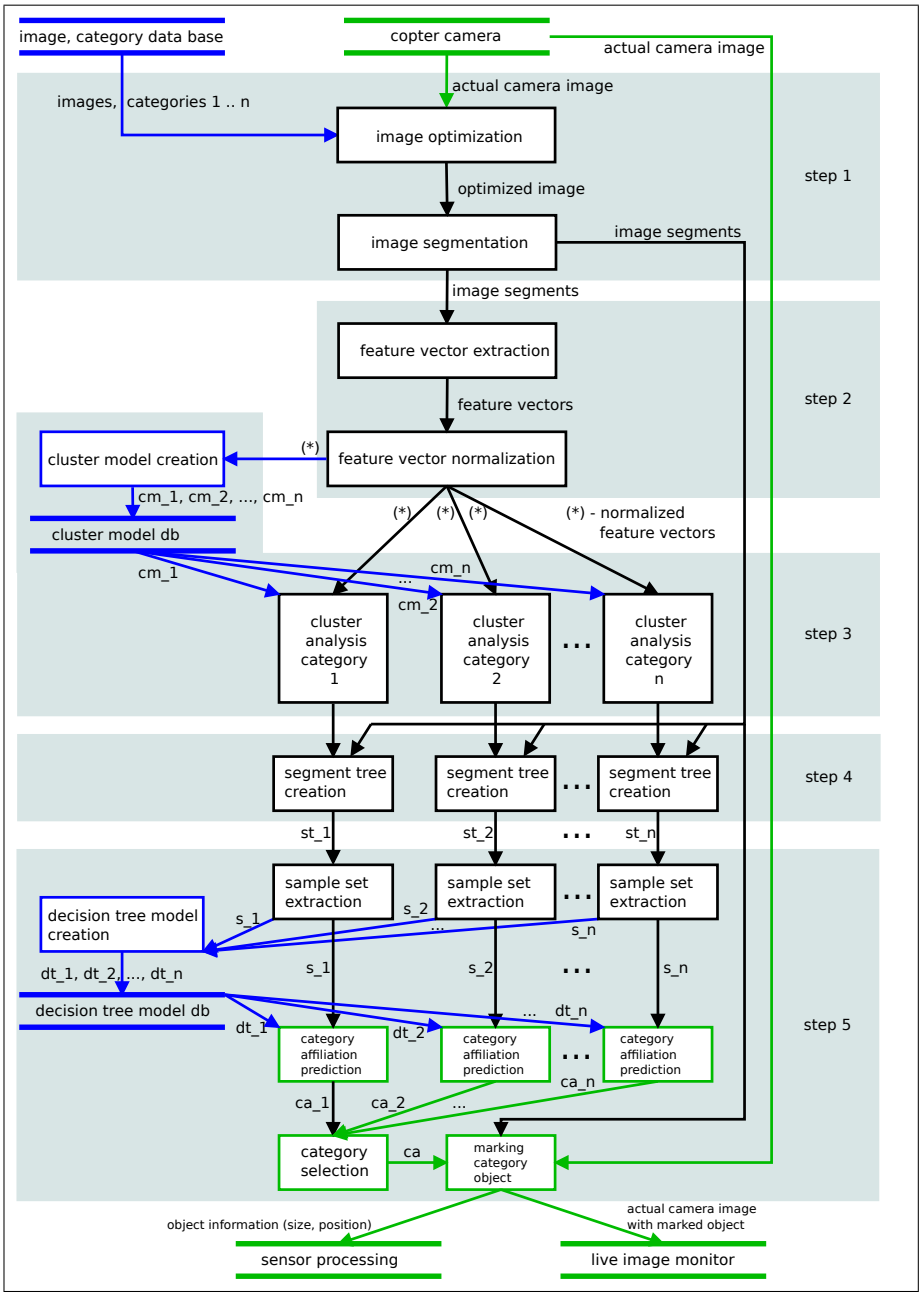

Fig. 1. Data flow diagram for the semantic object recognition procedure for training phase (blue) and recognition phase (green) with five steps: 1. image optimization and segmentation; 2. segment feature vector extraction; 3. compute/use cluster models over the feature vectors; 4. segment tree creation; 5. compute/use the decision tree models.

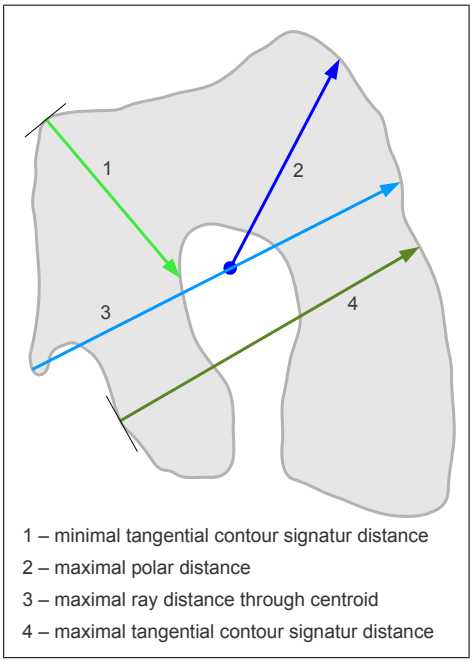

Fig. 2. Distance histogram methods: A distance histogram consists of a vector, where each element contains the distance between the centroid of the segment, strictly speaking the center of gravity, and a pixel in the segment contour (maximal polar distance) or the distance between two pixels in the segment contour computed with different methods (cf. [13, Sect. 3.1]).

3 Segment Contours and Perspective

Let us now consider the problem of perspective distortion in more detail, where objects are viewed from different viewpoints. In this context, the shape of an object is more or less defined by its surface. For the sake of simplicity, we assume that the surface is given as a polygon mesh. One question then is, what happens with the contour of a polygonal segment on the object surface, when it is viewed from different perspectives. Here, we restrict attention to the case where the segment is completely visible and not partially hidden. Let us consider the projective image of a cube as an example (Fig. 3). Clearly, by perspective projection, angles between lines and lengths of lines including their ratios may deviate significantly from their original values. In addition, parallelism of lines is not preserved, too.

Formally, the image of an object can be roughly described by central projection. Central projection (planar 3D projection) is determined in essence by the point of the observer, called central or focal point c and the position of the image plane, in particular its normal vector v, both consisting of 3D coordinates.

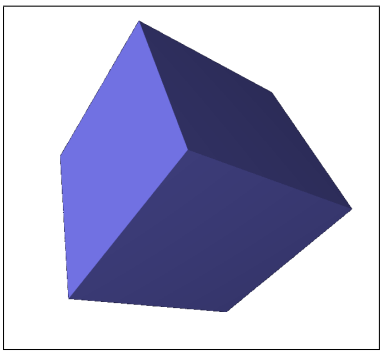

Fig. 3. Cube in perspective view.

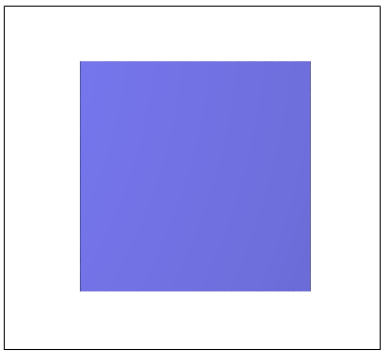

Fig. 4. Cube viewed from the front.

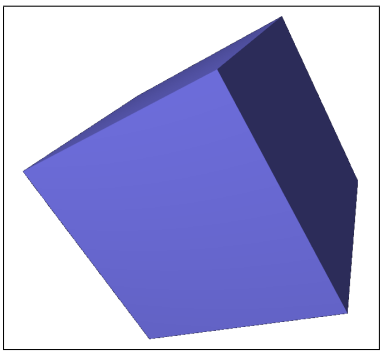

Fig. 5. Rotated cube.

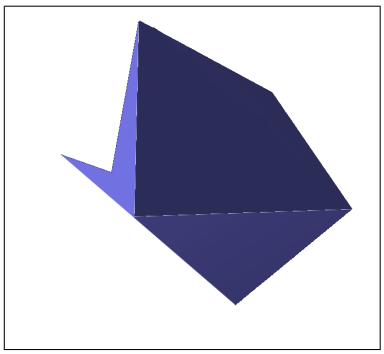

Fig. 6. Impossible projective view.

This gives us approximately 6 parameters. In contrast, a polygonal surface with n vertices is determined by n 2D coordinates, giving us $2n$ values. Since $2n > 6$ for $n \geq 4$, it follows that, although the shape of a polygon may vary widely by perspective distortion (cf. Fig. 4 and 5), for polygons with 4 or more vertices, there are clear limits for the distortion.

In fact, there are several invariants during perspective projection: First of all, straight lines remain straight lines. This implies that polygons remain polygons with the same number and order of vertices. Furthermore, left-right relations, which are important for localization, navigation and exploration [16], stay invariant, provided that the polygons are always viewed from the same side, which is usually the outside of the object. From this it follows in particular, that convex edges remain convex and never become concave by perspective projection, and vice versa. Therefore, the image in Fig. 6 definitely cannot be the projection of a cube, because the leftmost polygon in the image is concave, whereas a cube has only squares on its surface that are convex. Last but not least, adjacent segments remain adjacent. Hence, the image segment tree remains the same, although of course not all polygonal segment faces may be visible from all viewpoints.

In principle, full perspective projection can be taken into account: For this, a point x of a segment contour, given as 3D column vector in homogeneous coordinates, i.e. with an additional component (cf. [7, Sect. 5.6]), is projected onto the 2D plane, at position y, that is a 2D column vector (also in homogeneous coordinates). The corresponding mapping $M_0 : x \mapsto y$ consists first of a rotation R, that is 3×3 orthonormal matrix, and a translation T, a 3D column vector, and then the actual projection P from distance $-d$, as follows (cf. [7, Sect. 6.4]):

$$M_0 = \underbrace{\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1/d & 1 \end{bmatrix}}_{P} \cdot \begin{bmatrix} & R & & T \\ & & & \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Let now y_1 and y_2 be the projections of a given object point x, whose position usually is not known, on two images. Therefore, we have $y_i = M_i \cdot x$ for $i = 1, 2$, where M_1 and M_2 are mappings as above, in general different. From this, we obtain (1) $\lambda\, y_2 = M \cdot y_1$ with $M = M_2 \cdot M_1^{\dagger}$, where $\dagger$ denotes the Moore-Penrose pseudoinverse operator and λ a scale factor, which is needed because the 3×3 matrix M, mapping the homogeneous 2D coordinates, is not affine in general, i.e., its last row may be different from $[0 \cdots 0\ 1]$. Eq. (1) can be expressed equivalently without reference to λ by the cross product $y_2 \times (M \cdot y_1) = 0$. This leads to three linear equations in the 9 components of the matrix M, of which only two are linearly independent however, for each projection point pair (y_1, y_2). Given n such pairs, we arrive at the matrix equation $A \cdot m = 0$, where A is a $(2n) \times 9$ matrix and m is the matrix M reshaped as column vector (cf. [17]).

Since A is overdetermined in general and it must be $m \neq 0$, the solution for $A \cdot m = 0$ can be found by minimizing the squared (L2) vector norm (2) $\varepsilon = |A \cdot m|^2$ with respect to the condition $|m| = 1$. Eq. (2) is equivalent to $\varepsilon = (A \cdot m)^{\top} \cdot (A \cdot m) = m^{\top} \cdot (A^{\top} \cdot A) \cdot m$, where $\top$ denotes transposition, thus $(A^{\top} \cdot A) \cdot m = \varepsilon\, m$ (after multiplication with m). Hence, the problem of determining whether two segment contours stem from the same object, taking perspective distortion into account, can be reduced to an eigenvalue problem. As distance measure in the clustering procedure, the smallest eigenvalue ε of the 9×9 matrix $(A^{\top} \cdot A)$ can be used. Nevertheless, a normalization with respect to the starting point of the segment contour has to be done.

4 Evaluation of the Approach

The object recognition method with distance histograms (as described in Sect. 2) has been implemented in *C++/OpenCV* [4] by the first author. For the full treatment of perspective distortion (Sect. 3), so far only an implementation in *Matlab/Octave* [8] by the second author is available. All in all, our object recognition procedure works in practice: The segment neighborhood relations in the image segment tree remain invariant, even after perspective distortion. Rotation and translation of polygons is treated by length normalization. Although projections may vary a lot, they often show still strong similarity to the original

image. The clustering of different segments takes this into account. The experiments with our *C++* implementation for object recognition with recognition rates usually between 50 and 100 % – far above chance – are encouraging in this direction.

To test and improve the first implemented algorithm in a controlled environment, it was used to classify images from the butterfly image dataset [11]. For all seven categories, the right category of an image is predicted with a success rate of 99.5 % if the image is from the training set and 27.14 % if the image is from the test set. A random guess would give us only a success rate of $1/7 = 14.28\,\%$. On images made by the first author the success rates were 100.00 % and 46.00 % (5 categories). Here, a random guess would have a success rate of $1/5 = 20\,\%$ only. It takes about 0.7 seconds to classify a live image, which need not be pre-segmented into foreground and background.

5 Related Works

The problem of recognizing and locating objects is very important in applications such as robotics and navigation. Therefore, there are numerous related works. The survey [6] reviews literature on both the 3D model building process and techniques used to match and identify free-form objects from imagery, including recognition from 2D silhouettes.

[12] presents an object recognition system that uses local image features, which are invariant to image scaling, translation, and rotation, and partially invariant to illumination changes and affine or 3D projection. This proposed model shares properties with the object recognition in primate vision. A nearest-neighbor indexing method is employed that identifies candidate object matches. This approach is very successful in practice. However, as already said in the introduction, here more or less only points of clouds and not groups of segment faces are considered, which appears to be more cognitively adequate.

[14] performs shape retrieval by considering qualitative relations. This means the qualitative relations of the line segments forming the contour of the polygon are considered during the object recognition phase. The approach is eventually based on the so-called double-cross calculus [18]. Each object is identified by exactly one contour built from more and more polygon vertices. The approach is successful, however it does not reflect the fact, that complex objects may consist of several segment contours.

6 Conclusions

With our approach, we can recognize objects in digital images, independent of scaling, translation, rotation, and perspective distortions of the object. To do this, we train and use models based on the segment shapes of the objects and the topological and spatial relations of these segments. The next step is to implement the approach as a real-time object recognition process on autonomous multicopters (cf. [13]).

References

1. Alegre, E., Alaiz-Rodríguez, R., Barreiro, J., Ruiz, J.: Use of contour signatures and classification methods to optimize the tool life in metal machining. Estonian Journal of Engineering **1** (2009) 3–12
2. Attneave, F.: Some informational aspects of visual perception. Psychological Review **61**(3) (1954) 183–193
3. Berry, M.J.A., Linoff, G.: Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management. 3rd edn. John Wiley & Sons Inc. (2011)
4. Bradski, G.R., Kaehler, A.: Learning OpenCV – computer vision with the OpenCV library: software that sees. O'Reilly (2008)
5. Breiman, L., Friedman, J.H., Olshen, R.A., Stone, C.J.: Classification and Regression Trees. The Wadsworth Statistics/Probability Series. Wadsworth Publishing (1983)
6. Campbell, R.J., Flynn, P.J.: A survey of free-form object representation and recognition techniques. Computer Vision and Image Understanding **81**(2) (2001) 166–210
7. Foley, J.D., van Dam, A., Feiner, S.K., Hughes, J.F.: Computer Graphics: Principles and Practice. 2nd edn. The Systems Programming Series. Addison-Wesley Publishing Company (1993)
8. Gilat, A.: MATLAB: An Introduction with Applications. 2nd edn. John Wiley & Sons (2004)
9. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning: Data Mining, Inference, and Prediction. 2nd edn. Springer Series in Statistics. Springer (2009)
10. Jähne, B.: Digital Image Processing. 6th revised and extended edn. Springer (2005)
11. Lazebnik, S., Schmid, C., Ponce, J.: Semi-local affine parts for object recognition. In: Proceedings of the British Machine Vision Conference. Volume 2. (2004) 959–968
12. Lowe, D.G.: Object recognition from local scale-invariant features. Computer Vision, IEEE International Conference on **2** (1999) 1150–1157
13. Schmidsberger, F., Stolzenburg, F.: Object recognition with multicopters. In Wölfl, S., ed.: Poster and Demo Track of the 35th German Conference on Artificial Intelligence (KI–2012), Saarbrücken (2012) 83–87 Obtained Best Poster and Demo Presentation Award.
14. Schuldt, A.: Shape retrieval with qualitative relations: The influence of part-order and approximation precision on retrieval performance and computational effort. In Bach, J., Edelkamp, S., eds.: KI 2011: Advances in Artificial Intelligence – Proceedings of the 34th Annual German Conference on Artificial Intelligence. LNAI 7006, Berlin, Springer (2011) 301–312
15. Shuang, F.: Shape representation and retrieval using distance histograms. Technical report, Dept. of Computing Science, University of Alberta (2001)
16. Stolzenburg, F.: Localization, exploration, and navigation based on qualitative angle information. Spatial Cognition and Computation: An Interdisciplinary Journal **10**(1) (2010) 28–52
17. Wikipedia: Projektionsmatrix – Wikipedia: Die freie Enzyklopädie (2013) [Online; Stand 14. August 2013].
18. Zimmermann, K., Freksa, C.: Qualitative spatial reasoning using orientation, distance, and path knowledge. Applied Intelligence **6** (1996) 49–58

Challenges in Using Semantic Knowledge for 3D Object Classification

Corina Gurău[1] and Andreas Nüchter[2]

[1] Automation Group, Jacobs University Bremen gGmbH, Germany
[2] Robotics and Telematics, University of Wüzburg, Germany

Abstract. To cope with a wide variety of tasks, robotic systems need to perceive and understand their environments. In particular, they need a representation of individual objects, as well as contextual relations between them. Visual information is the primary data source used to make predictions and inferences about the world. There exists, however, a growing tendency to introduce high-level semantic knowledge to enable robots to reason about objects. We use the Semantic Web framework to represent knowledge and make inferences about sensor data, in order to detect and classify objects in the environment. The contribution of this work is the identification of several challenges that co-occur when combining sensor data processing with such a reasoning method.

1 Introduction

Autonomous recognition of structure in an indoor environment is a challenging task for the robotics community. Relying on depth perception, prior knowledge and logic, humans are particularly adroit at understanding their surroundings. Robotic systems rely on imagery and sensor data to build and encode their knowledge. Yet, we expect some systems to perform tasks such as navigation, manipulation, or interaction, in cluttered environments, structured for humans. To improve the way robots structure their knowledge of the world, we can share a common knowledge management system. Then robots could use our way to represent, make inferences and take decisions. By finding a representation in Description Logic for common-sense statements, and mapping them to ontological concepts and relations between those concepts, information such as *the book is on the shelf* or *the room is empty* is shared between humans and robots. This high level semantic description through ontologies also permits reasoning in a logical way.

In this paper we aim at verifying if the bottom-up, knowledge-based interpretation of indoor scenes is a reliable approach for 3D object detection. This task has been heavily performed using statistical methods and pattern recognition. Detecting and classifying objects by relying on a logical representation has been less considered in recent years, due to the access to large amounts of data and computational resources to learn the structure of our visual world.

Our proposed system is used for knowledge modeling and information retrieval. We divide the task into three main components: 1) geometric analysis

and characterization of scanned environment data, 2) semantic description and ontology mapping of geometric shapes, and 3) knowledge query and rule evaluation.

After scanning the environment, we use 3D point cloud segments to identify predefined geometric primitives and formalize spatial relations between object parts (cf. Fig. 1). We store the obtained geometric information and load it in a knowledge management system to populate an ontology with class instances. For answering queries over the computed spatial data, we implement a reasoner in Semantic Web Rule Language (SWRL) and run it under the platform of Protégé, an ontology editor and knowledge-base framework.

Space and spatial organization are the most common sense knowledge for humans. To describe them, we make use of Web Ontology Language (OWL). In our approach, we create an OWL ontology based on Description Logic (DL), which permits defining instances (description logic individuals), creating classes (description logic concepts), properties (binary relation specifying class characteristics), and operations (union, intersection, complement, etc). Our framework relies on reasoning with the 3D geometric information to detect and classify objects in a human environment. We consider properties such as size, orientation, position of point cloud segments, as well as spatial relations between segments, such as intersection, inclusion or parallelism. Our intuition in selecting the features is that it is easier to compute spatial relations for simple planar primitives of a complex object rather than computationally expensive ones for the whole object.

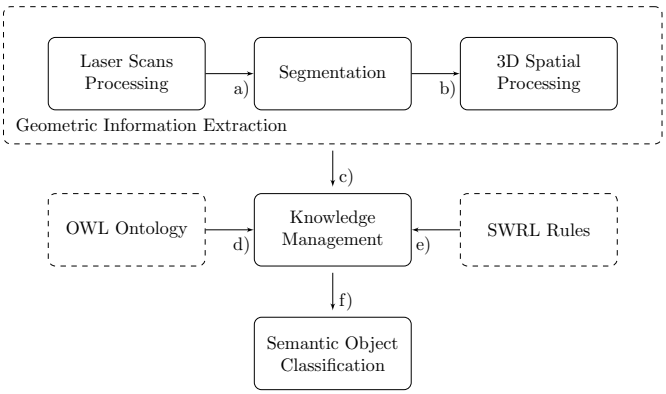

Fig. 1. Semantic Interpretation Pipeline. (a) Acquired laser scans are first segmented into planar regions. (b) Segments and spatial relationships are further analyzed. (c) Extracted geometric information is loaded in Protégé, on which our Knowledge Management framework is built. (d) An OWL Ontology encodes prior knowledge. (e) SWRL rules represent the restrictions we impose on the objects. (f) By running a reasoner on the information from (c), (d) and (e), objects of interest in the environment are detected and classified.

Related work in the area of combining 3D point cloud processing with knowledge-based reasoning is concerned with architectural reconstructions [2, 3]. A similar 3D object classification approach was taken by [4], however at critical points, the paper does not formulate solutions. In this paper, we focus on identifying the challenges in such an approach.

For the preprocessing phase we use the Felzenszwalb and Huttenlocher segmentation algorithm. Recently, we presented a segmentation method for 3D point clouds acquired with state-of-the-art 3D laser scanners extending the method of Felzenszwalb and Huttenlocher [1]. From the 3D points an unoriented graph is constructed. The graph is then segmented by using a k-nearest neighbor search and a similarity measure based on surface normals, resulting in a point cloud segmentation in planar patches. Fig. 2 shows two examples.

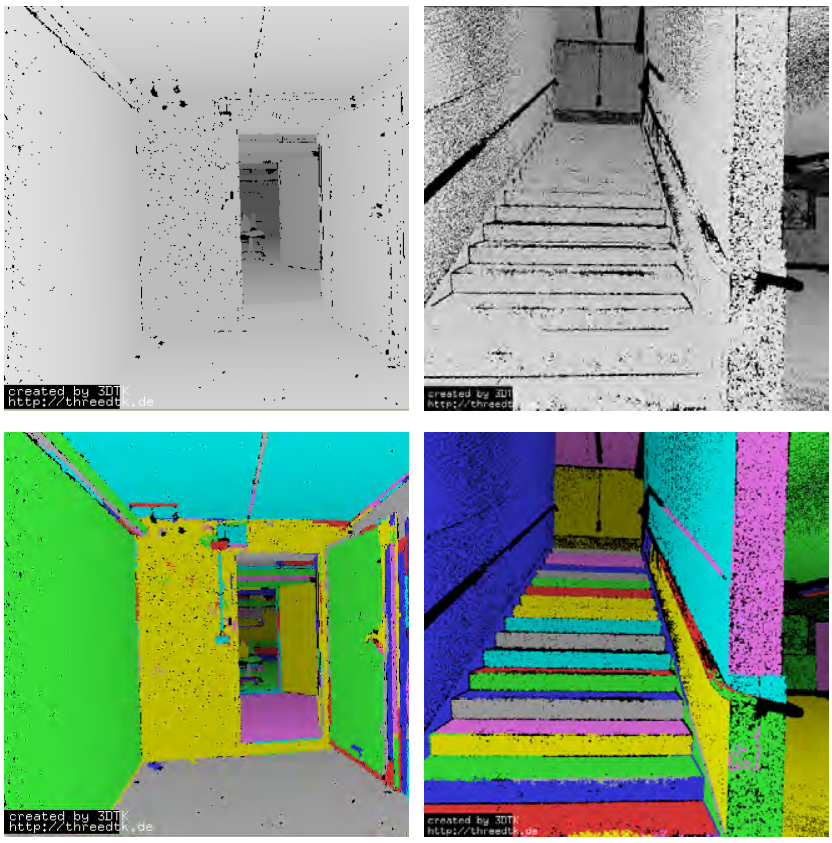

Fig. 2. Felzenszwalb and Huttenlocher Segmentation. Top: 3D Point Cloud of an empty room and of a staircase. The scene has been rendered with black fog to enhance depth perception. Bottom: 3D Point Cloud segmented using the parameters $\mu = 0, \sigma = 1$, $N = 10$, $k = 1000$.

2 The Protégé platform and Ontology Web Language

We model in an ontology our prior knowledge of the environment, making use of the Protégé-OWL editor. Protégé-OWL is an extension of Protégé that permits loading and saving ontologies, define logical class characteristics as OWL expressions, and most importantly, execute reasoners such as description logic classifiers. To complete the modeling process we add semantic rules developed with Semantic Web Rule Language (SWRL) and run Pellet, a Description Logic Reasoner, designed to work with OWL. Pellet is an implementation of a full decision procedure for OWL-DL which provides support for reasoning with individuals (asserted or inferred), user-defined datatypes and debugging and comparing ontologies.

Objects of interest in the scene are modeled under the class BuildingObject, while the rest map to geometries: either point cloud segments or pairs of point cloud segments. We therefore restrict our definition of an object to anything composed of them.

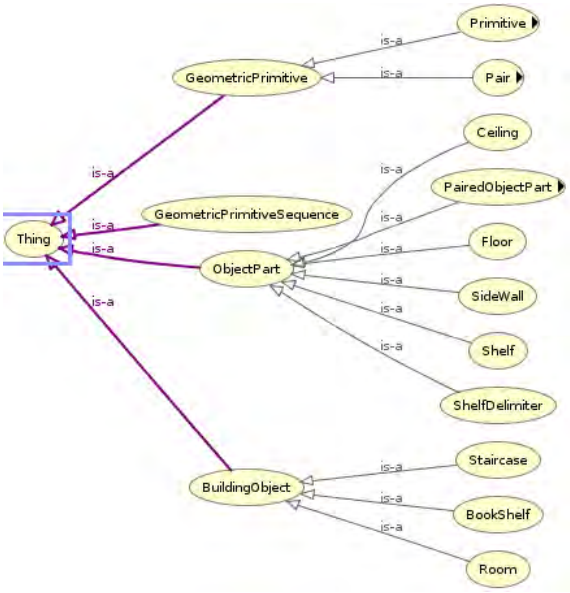

Fig. 3. Classes used for semantic interpretation.

Within the OWL ontology, not only we create appropriate object classes, but also class properties, through which we encode object geometry and spatial relations between segments in the scene (cf. Fig. 3). To integrate 3D data processing with Semantic Web technologies, we considered attributes such as:

size (Since we are only considering planar surfaces, we refer to size as the area of the segment. It is the most distinguishable segment property.), position (We consider minX, maxX, minY, maxY, minZ, maxZ as some objects are expected at a certain relative position inside a scene.), orientation (Individuals of vertical or horizontal segments are directly instantiated under the appropriate class.). Equally important as segment attributes, are the spatial relations between segments: connected, parallel, perpendicular, the pairs being instantiated under the classes Pair or PairedObjectPart.

3 SWRL Rules

The purpose of our semantic interpretation approach is to enable querying the spatial knowledge base. After populating our Protégé classes with individuals, we see their properties and their relationships as logical predicates (*asserted knowledge*), and we use logical rules to derive new facts and instances (*inferred knowledge*). The SWRL rules incorporate the restrictions that we impose on the environment: our knowledge about the scene configuration and about the shape of the objects. A rule takes the form of an implication between an antecedent and a consequent, and supports either a final decision or an intermediate decision in interpretation process. For instance, we know that a bookshelf essentially consists of a series of parallel segments at certain intervals. We make a similar judgement that if we have two stairs in the same sequence of primitives, the object is a staircase. Two example rules are as follows:

$$\begin{aligned}\text{LowShelf}(?x) \rightarrow\ & \text{HorizontalSegment}(?x) \wedge \text{hasSize}(?x, ?size) \\ & \wedge \text{swrlb} : \text{greaterThan}(?size, 0.02) \wedge \text{swrlb} : \text{lessThan}(?size, 1.0) \\ & \wedge \text{hasMaxY}(?x, ?maxY) \wedge \text{swrlb} : \text{greaterThan}(?maxY, 0.6) \\ & \wedge \text{swrlb} : \text{greaterLess}(?maxY, 1.5)\end{aligned}$$

$$\begin{aligned}\text{Staircase}(?x) \rightarrow\ & \text{hasHVConnectedPair}(?x, ?pair1) \wedge \text{Stair}(?pair1) \\ & \wedge \text{hasHVConnectedPair}(?x, ?pair2) \wedge \text{Stair}(?pair2) \\ & \wedge \text{GeometricPrimitiveSequence}(?x)\end{aligned}$$

4 Results

To show the potential of our approach we exhibit three different simulations in which we query the knowledge system for different building objects. Our approach is also viable for different geometries, in particular after extending the method to curved spaces by adding properties and rules accordingly.

Our simulations concern half of an empty room, a staircase and a bookshelf. For each scenario, a set of SWRL rules was designed that allows for labeling of intermediate object parts such as a ceiling, shelf planes or stairs, as well as labeling of the entire object of interest. Labels correspond to object categories. We

map the segmentation output to the ontology via a mapping language, and obtain asserted instances. By running the reasoner, we further label the segments, and create inferred instances. For the three examples, the results are shown in Table 1. Not all mapped segment get a labeling, which is due to the challenges described next.

Table 1. Asserted and inferred segment labels

point cloud	#mapped segments	#labeled segments
empty room	9	5
bookshelf	18	18
staircase	17	5

5 Challenges

Several challenges were encountered during the implementation of the presented object classification method. More precisely, we coped with:

Thresholds. As the description logic reasoner uses crisp logic, we had to set hard thresholds for property predicates, e.g. swrlb : greaterLess(?maxY, 1.5). Finding these constants was done manually and it was time-consuming.

Coordinate frames. In our experiments the constants refer to the scanner own coordinate system and we used single scans. It is an open question how this extends to an arbitrary (project or robot) coordinate system or even to global coordinates, i.e., to include georeferencing.

Missing data. We experienced that mapped segments are not labeled due to missing data. The laser scanner gages only objects visible. However, multiple 3D scans and scan registration are necessary to completely digitalize scenes.

Efficiency for multi-values predicates. For extracting relations between individuals they have to be compared. Currently, we perform this comparison while processing the point cloud in C++, exploiting spatial data structures such as k-d trees.

Memory efficiency. Due to the presence of many segments in realistically sized real-world scenes, Pellet reasoner tends to run out of memory due to the complexity of the used description logic.

Designing the data processing tool chain. It is not clear, which parts of the interpretation process should be implemented at the point cloud processing level, i.e., in the C/C++ part that acquires the sensor data, calculates the normals and performs the segmentation, and which parts should

be performed by description logic reasoning in the knowledge-based system. The question is, when and where to call Pellet and the used ontology.

6 Conclusion

We presented a framework for semantic interpretation of point clouds which takes advantage of Semantic Web technologies. Built on the platform of Protégé-OWL, our alternative method of linking top level semantic qualification with low level geometric calculations uses a connectivity-preserving segmentation algorithm, an ontology structure and a reasoner. We believe that the logical structure of an ontology is suitable for semantic knowledge representation and that under the Semantic Web framework, Web Ontology Language is appropriate for defining spatial knowledge. Such an approach provides a better understanding of a 3D scene, by facilitating detection and recognition in 3D point clouds.

Needless to say, a lot of work remains to be done. To avoid the use of crisp thresholds, we plan to add fuzziness to the system and/or use probabilistic reasoning. A promising approach is given by Pu and Vosselmann in [5]. They use semantic building knowledge to reconstruct a polyhedron model of outdoor terrestrial 3D scans. They also describe the uncertainty and make expected decisions [6]. Further future work will aim at interpreting multiple registered 3D scans. As our system relies on plane segmentation, this extension seams straightforward. However, a combination with next-best-view planning is highly desirable.

References

1. Sima, M.C., Nüchter, A.: An extension of the Felzenszwalb-Huttenlocher segmentation to 3D point clouds. In: International Conference on Machine Vision (2012)
2. Duan, Y., Cruz, C., Nicolle, C.: Architectural reconstruction of 3D building objects through semantic knowledge management. In: 11th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (2010)
3. Hmida, H., Cristophe, C., Frank, B., Christophe, N.: Knowledge Base Approach for 3D Objects Detection in Point Clouds Using 3D Processing and Specialists Knowledge. International Journal On Advances in Intelligent Systems, vol. 5, pp. 114 (2012)
4. Günther, M., Wiemann, T., Albrecht, S., Hertzberg, J.: Model-based object recognition from 3D laser data. KI 2011: Advances in Artificial Intelligence, Springer (LNAI 7006), pp. 99-110 (2011)
5. Pu, S., Vosselman, G.: Knowledge based reconstruction of building models from terrestrial laser scanning data. {ISPRS} Journal of Photogrammetry and Remote Sensing **64**(6), 575 – 584 (2009)
6. Pu, S.: Knowledge based building facade reconstrcution from laser point clouds and images. PhD thesis, University of Twente (2010)

KI 2013 Workshop on Visual and Spatial Cognition

Exploring Spatio-Temporal Data Modeled as Dynamic Weighted Relations

Michael Burch, Michael Raschke, Daniel Weiskopf

VISUS, University of Stuttgart
Allmandring 19, 70569 Stuttgart, Germany

Abstract. Eye tracking studies lead to spatio-temporal data in the form of gaze trajectories that show the behavior of gaze positions over time. Such data can be modeled as a dynamic graph that expresses the transitions of gaze positions between Areas of Interest (AOIs) by time-varying weighted relations. Moreover, a hierarchical organization of the AOIs may be of interest, resulting in a dynamic compound AOI digraph. Traditionally, this kind of time-based relational data is represented by animated node-link diagrams that are laid out with respect to a list of aesthetic graph drawing criteria. In our work, we propose a visual metaphor for displaying relational data that uses space-filling circle sectors to encode dynamic relations between hierarchy elements. The idea benefits from the fact that dynamic compound digraphs can be visualized with reduced visual clutter compared to node-link diagrams for dense graphs. Finally, we illustrate how interaction methods can be used to explore a dataset for trends, countertrends, and/or anomalies.

1 Introduction

Information hierarchies are present in many application domains such as in the hierarchical organization of software and river systems. Also, Areas of Interest (AOIs) are useful when exploring spatio-temporal eye tracking gaze trajectories, which can be hierarchically organized. Moreover, the eye movement behavior of study participants has a spatio-temporal nature. By using the AOI information and the gaze trajectories this data can be modeled as a dynamic weighted compound AOI digraph. Consequently, this kind of relational data can be visually explored with the general concept proposed in this paper.

The efficient representation of hierarchical data has been in focus of information visualization research ever since. Hierarchies are, for example, displayed by traditional node-link diagrams [13], Treemaps [8], indented plots [5], or layered icicles [9].

General graphs—if they do not belong to the class of planar graphs—suffer from visual clutter [14] when depicted graphically by a naive layout technique. Consequently, over the years many sophisticated graph drawing algorithms were developed to make graphs aesthetically pleasing [10–12, 16]. Apart from reducing edge crossings—which is the major criterion in graph drawing—reducing edge lengths, maximizing orthogonality and symmetries, or minimizing display space

are other aesthetic criteria among a larger set. Typically, graph data does not stay static but changes over time, leading to much research in this domain [2, 3, 6, 7].

In this paper, we propose an interactive visualization tool for representing and manipulating this kind of dynamic graph data. Our approach uses a radial visual metaphor and is based on the visualization principles proposed by Burch and Diehl in their TimeRadarTrees technique [3]. In particular, eye tracking data is of spatio-temporal nature and is visualized by heatmaps or gaze plots as traditional concepts. Heatmaps are time-aggregated representations only showing the hot spots, whereas gaze plots suffer from visual clutter. Also, AOI rivers [4] produce many crossings in the display but better show the temporal evolution of eye gaze frequencies between AOIs in displayed stimuli.

The work of Blaschek and Ertl [1] describes an approach that is useful for supporting researchers working in the field of information visualization when deriving insights from eye tracking experiments. The proposed framework uses visual analysis methods to evaluate eye tracking data. Our proposed method transforming eye movement data to dynamic graphs might be used as one candidate of such an analysis method among others.

We illustrate the visualization tool in a stepwise manner and follow the visual information-seeking mantra [15] that summarizes many visual design guidelines and provides an excellent framework for designing information visualization applications. The visual information-seeking mantra divides the visual exploration process into the following three stages: Overview first, zoom and filter, then details-on-demand.

2 Dynamic Graph Visualization

The main focus of our visualization tool is the visualization of dynamic relational data in information hierarchies as static pictures that can be manipulated and analyzed interactively. This representation stands in contrast to animation-based techniques for displaying dynamic data. The other visualization views presented here support users when they want to obtain an overview of the dataset or zoom in and apply filtering functions to the dataset.

2.1 Data Model

We model an information hierarchy as a cycle-free and connected graph $H = (V, E)$, where V is the set of nodes and $E \subsetneq V \times V$ expresses the set of directed edges—the link information in the hierarchy. $L \subsetneq V$ is the set of leaf nodes in the containment hierarchy. All other nodes $V \setminus L$ are containers that hierarchically bundle this information. The leaf nodes can be related to some extent and actually form a directed graph with edge weights $G = (L, E_G)$, where $E_G \subseteq L \times L$ and $w_G(u, v) \in \mathbb{R}_0^+$ denotes the weight function for $u, v \in L$.

A sequence of graphs can be modeled by $G_i = (L_i, E_{G_i})$ where $E_{G_i} \subseteq L_i \times L_i$. $w_{G_i}(u, v) \in \mathbb{R}_0^+$ is the weight function for $u, v \in G_i$. If the context is clear, we omit the graph G and use $w_i(u, v)$ instead of $w_{G_i}(u, v)$.

A set of gaze trajectories as generated during eye tracking experiments can be transformed into dynamic weighted directed graphs by subdividing the time into subintervals each corresponding to one graph in the sequence. The AOI information can be modeled as graph vertices and the number of eye movements between pairs of AOIs as directed weighted edges. If a hierarchical organization among the AOIs exists or one can be computed by a hierarchical clustering algorithm, then this spatio-temporal eye tracking data can be transformed into a dynamic weighted compound AOI digraph.

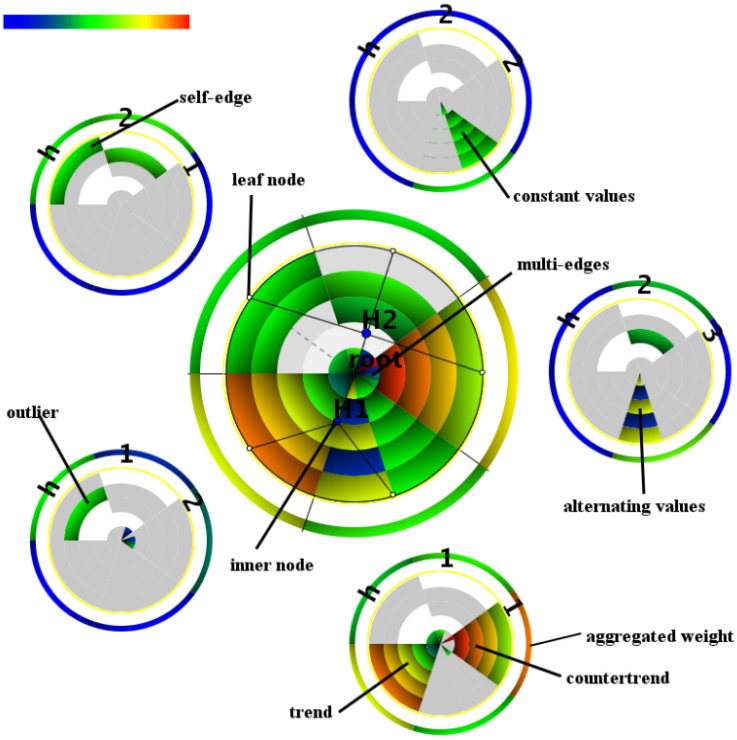

Fig. 1. A sequence of five compound directed graphs in a TimeRadarTrees representation with color-coded edge weights.

2.2 Visual Encoding

For displaying the dynamics of relations in information hierarchies, we use a TimeRadarTrees [3] visualization. Figure 1 shows a sequence of five directed and weighted graphs in this visual metaphor. The visualization is an integration of three views into one.

- **Hierarchy View:** The hierarchy of the selected elements is represented as a radial node-link diagram with the root node in the circle center and the node sizes and colors depending on the size and depth of their subtrees.
- **Time Radar View:** The large circle in the center is used to represent the sequence of graphs. Each circle slice encodes a single graph and each circle sector one specific node in this sequence. A weighted edge is represented by a color-coded circle sector. The circle is divided into as many sectors as different nodes are present in the graph sequence. The time axis starts in the circle center and heads radially to the circumference.
- **Thumbnail View:** Thumbnails—the miniature representations outside the larger circle in the middle—can be used to derive start and destination nodes of a weighted edge. By inspecting the color, the shape, the curvature, the direction, and/or orientation of a small circle sector and comparing it to the larger one in the center circle, one can derive a relation between two nodes.

For the example in Figure 1, one can detect that the hierarchy consists of five leaf nodes, namely $h11, h12, h21, h22$, and $h23$. These are organized into two subhierarchies, $H1$ and $H2$, which are again direct children of the root node. Five graphs are represented, which can be inferred from the fact that each circle is divided into five color-coded slices. The color coding in use is a vegetation color scale that maps lower values to blue and higher ones to red. Values in between are colored from green to yellow. The outer smaller slice encodes the aggregation of edge weights over the whole graph sequence and can be explored for each node pair separately.

The upper left green-colored sector in the thumbnail of node $h21$ represents a self-edge—an edge that starts and ends at the same node. The permanently green-colored sectors in the thumbnail of node $h22$ show that this relation to node $h11$ always exists in the whole sequence and moreover, has a constant weight. The thumbnail of node $h23$ on the right hand side has an alternating relation to node $h11$. This phenomenon can be examined by the alternating color coding between blue and yellow. A trend and a countertrend are visible in the thumbnail of node $h11$, where the strength of the relation to node $h23$ decreases at a constant level. The same is true for the relation to node $h12$, but here we can detect an increase of the weight. An outlier may be the green sector in the thumbnail of node $h12$ that points to node $h21$ and hence into a different hierarchy level. It goes without saying that this kind of graph representation also allows multi-edges—two or more edges between the same two nodes in a graph. There are actually three edges in the first graph starting at node $h12$ and targeting to node $h23$.

3 Working with the Tool

Figure 2 shows the graphical user interface of the visualization tool, which is divided into several components. The upper left frame shows the overview by a scrollable pixel map. Below it, the selected part of the pixel map is shown

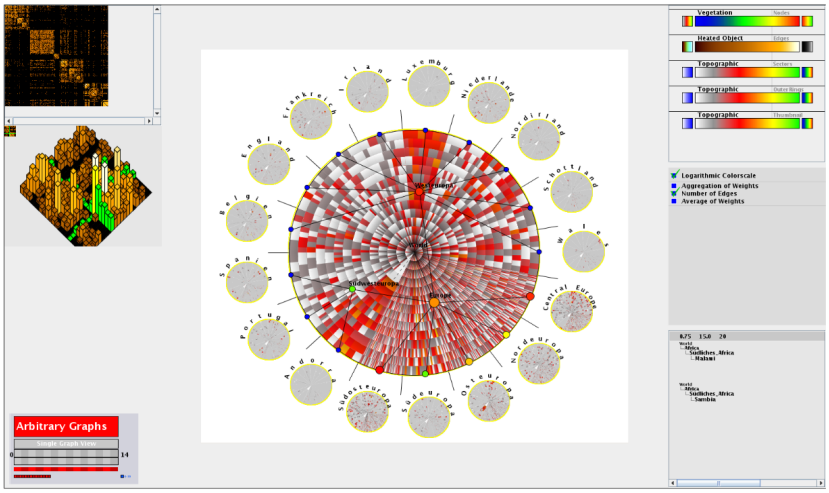

Fig. 2. The graphical user interface of the visualization tool is divided into several views. The TimeRadarTrees view in the center shows the evolution of relations in information hierarchies of the selected hierarchies and time period.

as a three-dimensional bar chart as some kind of zooming function. The filter functions are located in the bottom view on the left hand side. The three views at the right hand side are used for applying a special color coding, selecting an aggregation type or logarithmic scale, and a details-on-demand view. The main view in the center represents the TimeRadarTrees visualization for the selected nodes and time periods. In the following, we explain the single visualization components and interaction techniques to manipulate the data.

3.1 Overview: Pixel Map

As we follow the information seeking-mantra, we first give an overview of the whole relational dataset. To this end, we aggregate the graph sequence to one aggregated graph that we represent as a pixel map—a color-coded adjacency matrix. The color coding depends on the aggregation type that can be:

– **Weighted sum:** All weights in the graph sequence are summed up:

$$w_{agg}(u,v) := \sum_{1 \leq i \leq n} w_i(u,v) \; \forall u,v \in E_i$$

– **Average weight:** The weights are summed up and divided by the number of occurrences of this edge in the sequence:

$$w_{agg}(u,v) := \frac{\sum_{1 \leq i \leq n} w_i(u,v)}{\mid \{(u,v) \mid (u,v) \in E_i\} \mid} \; \forall u,v \in E_i$$

If the number of edges is zero, the weight is defined by

$$w_{agg}(u, v) := 0.$$

– **Number of edges:** The number of edges depends on the weight filter that is again defined by the minimum weight v_{min} and the maximum weight v_{max}. Hence, we define in this case

$$w_{agg}(u, v) :=\mid \{(u, v) \mid v_{min} \leq w_i(u, v) \leq v_{max}\} \mid$$

The users can interactively change the type of aggregation until they obtain interesting insights in the dataset. At this early stage, they can detect patterns and anomalies by exploring the hierarchically ordered pixel map. If they find interesting behaviors in the dataset, they may wish to filter the data for smaller subsets and obtain a three-dimensional bar chart of the brushed elements. Further examination of the selected part leads to more specific insights in the dataset that was not possible in the overview visualization.

3.2 Zoom: 3D Bar Charts

Figure 3 shows three-dimensional bar charts for the selected area in the pixel map. One big advantage of this representation is that we can encode two different metrics at the same time—one in the height and one in the color coding. This could help find out which relations occur very often and also have a very high aggregated weight. We cannot obtain this insight in the overview-based pixel map view apart from the fact that we may switch the aggregation type.

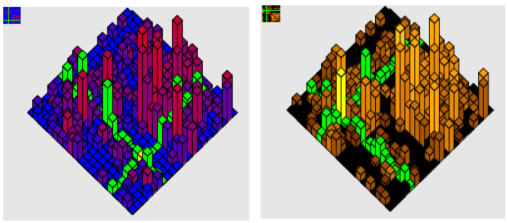

Fig. 3. Three-dimensional views of a selected subset can encode two metrics at the same time—one in the height of the bars and one in the color coding.

A crosshair function supports selecting a bar in the three-dimensional view. Selecting this bar can be difficult due to occlusion problems. The selected aggregated relation is given as a details-on-demand information with its metric values nearby.

3.3 Data Filtering

The tool is able to handle three different types of filtering functions that can be applied separately.

- **Weight filter:** First of all, one can apply a weight filter on the single values of each relation. Only those values are counted as relations that lie between the given thresholds. The aggregated values can be filtered for minimum and maximum sum, minimum and maximum average value, and minimum and maximum edge number.
- **Hierarchy filter:** The weight filter can be applied to the relations, another type of filter can be used to select a group of nodes from the hierarchy. Only those nodes are represented that belong to the selected group. This filtering technique is very important for the TimeRadarTrees visualization because it has a low scalability in this dimension.
- **Time filter:** Time is the third dimension where we can apply a filtering function. The thresholds can be given as an interval where all graphs are displayed that lie within this interval or single graphs can be selected and deselected again.

3.4 TimeRadarTrees

The main part of the visualization tool is the TimeRadarTrees representation in the center. It is used to display the dynamic graph data in information hierarchies in a static picture. Figure 1 illustrates the important visual signatures that are visible in the TimeRadarTrees representation for the example of a small dynamic graph. User interaction can be applied to modify the TimeRadarTrees view and to gain insights from the data on a dynamic level.

Interaction methods supported by the tool include filtering, adapting the color coding, aggregation of relations, collapsing and expanding subhierarchies, time warp, sector highlighting, textual search, changing of the hierarchy visual metaphor, cushion effect on the circle sectors, stacking of radial bar charts, etc.

4 Conclusion and Future Work

In this paper, we proposed a visualization tool for exploring dynamic compound digraphs in information hierarchies. Such datasets can be generated by taking spatio-temporal eye movement data and corresponding areas of interests into account. By doing this, a dynamic weighted digraph is produced that can then be visually explored by our dynamic graph visualization tool. In general, the difficulty is to first present the data in a pixel-based overview and allow the user to zoom and filter the dataset in several dimensions, i.e., weight, hierarchy, and time.

The TimeRadarTrees visualization technique is used to give insights in the dynamic data. The strength of this technique is the static representation of dynamic data that shows several dimensions in a single view—relation weights, hierarchical organization, and time periods. Interaction techniques support the user in obtaining insights in the dataset.

We plan to implement more interaction techniques such as a radial and a sector distortion. To this end, the TimeRadarTrees visualization will be mapped

to a circular shape. A sector distortion would allow unequal sized sectors, but the whole structure would still be mapped on a circle. As an enhancement, we could allow a distortion of the circular shape by dragging and dropping the circumference, thus obtaining an irregular shape that may improve the visual exploration when also applied to the thumbnail view in the same way.

The weakness of our approach—in contrast to node-link diagrams—is the lack of support for solving path-related tasks. To address this problem, we plan to implement an interaction method that highlights paths in the graph data displayed as TimeRadarTrees.

References

1. Blascheck, T., Ertl, T.: Techniques for analyzing empirical visualization experiments through visual methods. In: Proceedings of Workshop on Visual and Spatial Cognition (2013, to appear)
2. Burch, M., Beck, F., Diehl, S.: Timeline trees: Visualizing sequences of transactions in information hierarchies. In: Proceedings of 9th International Working Conference on Advanced Visual Interfaces. pp. 75–82 (2008)
3. Burch, M., Diehl, S.: TimeRadarTrees: Visualizing dynamic compound digraphs. Computer Graphics Forum 27(3), 823–830 (2008)
4. Burch, M., Kull, A., Weiskopf, D.: AOI rivers for visualizing dynamic eye gaze frequencies. Computer Graphics Forum 32(3), 281–290 (2013)
5. Burch, M., Raschke, M., Weiskopf, D.: Indented Pixel Tree Plots. In: International Symposium on Visual Computing. pp. 338–349 (2010)
6. Burch, M., Vehlow, C., Beck, F., Diehl, S., Weiskopf, D.: Parallel edge splatting for scalable dynamic graph visualization. IEEE Transactions on Visualization and Computer Graphics 17(12), 2344–2353 (2011)
7. Greilich, M., Burch, M., Diehl, S.: Visualizing the evolution of compound digraphs with TimeArcTrees. Computer Graphics Forum 28(3), 975–982 (2009)
8. Johnson, B., Shneiderman, B.: Tree-maps: A space-filling approach to the visualization of hierarchical information structures. In: Proceedings of IEEE Visualization Conference. pp. 284–291 (1991)
9. Kruskal, J., Landwehr, J.: Icicle Plots: Better displays for hierarchical clustering. The American Statistician 37(2), 162–168 (1983)
10. Misue, K., Eades, P., Lai, W., Sugiyama, K.: Layout adjustment and the mental map. Journal of Visual Languages and Computing 6(2), 183–210 (1995)
11. Purchase, H.C.: Which aesthetic has the greatest effect on human understanding? In: Graph Drawing. pp. 248–261 (1997)
12. Purchase, H.C., Cohen, R.F., James, M.I.: Validating graph drawing aesthetics. In: Graph Drawing. pp. 435–446 (1995)
13. Reingold, E.M., Tilford, J.S.: Tidier drawings of trees. IEEE Transactions on Software Engineering 7(2), 223–228 (1981)
14. Rosenholtz, R., Li, Y., Nakano, L.: Measuring visual clutter. Journal of Vision 7(2), 1–22 (2007)
15. Shneiderman, B.: The eyes have it: A task by data type taxonomy for information visualizations. In: IEEE Symposium on Visual Languages. pp. 336–343 (1996)
16. Ware, C., Purchase, H.C., Colpoys, L., McGill, M.: Cognitive measurements of graph aesthetics. Information Visualization 1(2), 103–110 (2002)

KI 2013 Workshop on Visual and Spatial Cognition

Techniques for Analyzing Empirical Visualization Experiments Through Visual Methods

Tanja Blascheck and Thomas Ertl

Institut for Visualization and Interactive Systems, University of Stuttgart
Universitätsstraße 38, 70569 Stuttgart, Germany

Abstract. Analyzing data collected in empirical visualization experiments is a time consuming task. In order to support visualization designers analyzing the data collected during such experiments, we intend to develop an analysis model. This analysis model will automatically choose appropriate visual analysis methods according to an analysis task to evaluate a new visualization technique. We will contribute an idea how this analysis model can look like, and what steps need to be taken to define the model. Furthermore, we investigate which visual analysis methods are available or have to be developed first, and how analysis tasks can be defined.

Keywords: Analysis Model, Visual Analysis Methods, Analysis Tasks, Empirical Visualization Experiment

1 Problem Description

Information visualization has the goal to represent information data in a graphical way to uncover concealed inner relationships by offering suitable visual representations. To evaluate if a new information visualization technique supports a user to uncover these relationships user experiments can be conducted [9]. A user experiment can for example be an eye tracking study where participants have to solve tasks with the new visualization technique. Such user experiments will be called empirical visualization experiments in this paper. The generic term, *empirical visualization experiment* means that every possibility to evaluate a new visualization techniques can be used.

The analysis process evaluating data collected in empirical visualization experiments is often a time consuming task. Furthermore, visualization designers often don't have the skills to conduct and analyze such experiments. Therefore, we contribute a concept for an analysis model which will support visualization designers to analyze collected data in an experiment. This analysis model will be based on *analysis tasks* and *visual analysis methods*.

An analysis task is derived from a research question, to specify how this research question can be validated. For example, if an empirical visualization experiment wants to compare two visualization techniques with each other and

find out why one visualization technique can be used to answer a task faster, a potential analysis task could be to investigate the "overall spatial pattern of [eye] movements" [1].

Visual analysis methods are visualization techniques designed or adapted to represent experimental data, such as for example eye movement data. The "Time Radar Tree" visualization technique developed by Burch et al. [3] can for example be used to explore spatio-temporal eye movement data modeled as dynamic weighted relations.

Therefore, our work on an analysis model aimt at answering the following research question: What are appropriate visual analysis methods for analysis tasks required to evaluate empirical visualization experiments? To answer this question, this paper will outline how an analysis model could look like and what steps need to be taken to develop such an analysis model.

2 Goal Description

Visualization research can be segmented into scientific visualization and information visualization. Scientific visualization uses data from domains such as biology, engineering, or meteorology, and is often inherently spatial. Information visualization maps abstract data to a spatial domain, such as for example data from social networks, or business data [7]. To evaluate if a new visualization technique supports a user empirical visualization experiments can be conducted [9].

In this paper, we will present an outline of an analysis model to support visualization designers during the evaluation of an empirical visualization experiment. Our analysis model (cf. middle block of Figure 1) uses input from three different data categories: information about the experimental design, the experiment conduction, and the matching model. These three data categories will be described in more detail in the following.

The experimental design is shown in the upper block in Figure 1 and can be classified by an experimental categorization based on the visualization technique evaluated, the experimental design method, the research question, and the data collection methods used. We will only evaluate visualization techniques from information visualization, such as node-link-diagrams, or time radar trees. Data collection methods can be interviews, cases studies, surveys, focus groups, data collections using eye tracking, or else. The research question investigated has to be defined by a user.

Possible data collected during the experiment conduction is shown in the left block in Figure 1. This data will be automatically collected and should be available in a machine readable format. The data types will depend on the data collection method defined in the experimental design and can for example be eye tracking data, participant data, benchmark data such as completion times and accuracy rates, and interaction data.

The most important part of our analysis model is the matching model in the lower block in Figure 1. It contains possible analysis tasks as well as visual

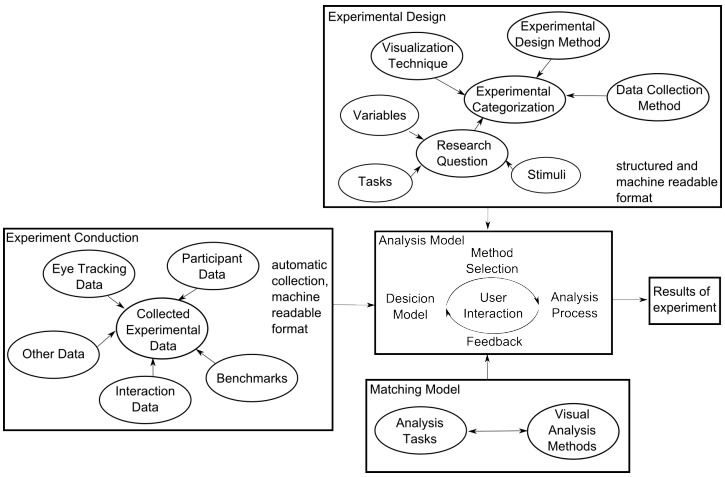

Fig. 1. The analysis model (middle rectangle) uses input data from the experimental design (upper rectangle), and experimental conduction (left rectangle). The matching model suggest appropriate visual analysis methods based on the analysis task (lower rectangle). The decision model suggests appropriate visual analysis methods to a user to create results for an experiment.

analysis methods. In the matching model each analysis task will be matched to one or multiple visual analysis methods.

The analysis model itself furthermore consists off a decision model and of the analysis process. The decision model suggests appropriate analysis tasks, and visual analysis methods based on the information from the experimental design. It will also create a chosen visual analysis methods based on the collected experimental data. The analysis process is the part where user is included and can give feedback. We have defined the following steps for the decision model (DM) and the analysis process (AP).

1. DM: Based on the information from the experimental design the decision model suggests multiple appropriate analysis tasks to evaluate the research question.
2. AP: The user chooses one analysis task he wants to investigate.
3. DM: Based on the decision of the user, several visual analysis methods are suggests appropriate for the analysis task.
4. AP: The user chooses one of the offered visual analysis methods.
5. DM: The chosen visual analysis method is created by the decision model using the collected empirical data.
6. AP: The created visual analysis method is used to interrogate the research question.
7. AP: Go back to step 1 or 3 and repeat until results are enough.

Creating this analysis model requires answering the following subquestions. For each subquestion we propose to take the following steps, which will be discussed further in section 3.

- Which empirical visualization experiments are being used to evaluate visualization techniques and how can they be classified? → Section 3.1: Categorization of empirical visualization experiments.
- Are there general analysis tasks for analyzing data collected in an empirical visualization experiment? → Section 3.2: Definition and evaluation of analysis tasks.
- Which visual analysis methods are available to analyze collected data in an empirical visualization experiment? → Section 3.3: Definition and evaluation of visual analysis methods.
- Which visual analysis methods are appropriate for each analysis task and how can visual analysis methods be mapped to analysis tasks in a general way? → Section 3.4: Creation and validation of matching process for analysis tasks with visual analysis methods
- Which new visual analysis methods first have to be developed? → Section 3.5: Creation and evaluation of new visual analysis methods.
- Is there a general analysis model to evaluate empirical visualization experiments and how can this look like? → Section 3.6: Creation and validation of the analysis model.

3 Method Description and Procedure

In the following, our methods and procedures are described for each step defined in detail in section 2.

3.1 Categorization of Empirical Visualization Experiments

Psychology differentiates between true experiments, quasi experiments, and non experiments. Data collection methods can for example be interviews, case studies, focus groups, or data collection using eye tracking. The first step, to find an analysis model for empirical visualization experiments will be to find categories for empirical visualization experiments, and to classify those depending on the experimental method, the data collection method, the type of visualization technique investigated, as well as the research question investigated. This categorization is necessary, to find appropriate analysis tasks. It will include a review of conducted empirical visualization experiments.

3.2 Definition and Evaluation of Analysis Tasks

Based on the categorization of empirical visualization experiments appropriate analysis task will be derived. This is based on related work, as well as on expert reviews with visualization designers conducting experiments. These expert

reviews are necessary to understand different types of visualization techniques, their goals and tasks, in order to infer appropriate analysis tasks. Evaluation of analysis tasks will be performed by visualization designers by conducting a user experiment, where participants have to match analysis task to types of visualization techniques.

3.3 Definition and Evaluation of Visual Analysis Methods

As a next step, an investigation of available visual analysis methods will be conducted. Here participants from the field of psychology or human-computer interaction who have conducted user experiments will be interviewed which visual analysis methods exists or which visual analysis methods are missing for analyzing empirical visualization experiments.

3.4 Creation and Validation of Matching Process for Analysis Tasks with Visual Analysis Methods

After defining analysis tasks, and collecting appropriate visual analysis methods the matching process will be developed, based on matching guidelines, which will be to be defined first. This matching process will be evaluated in a user experiment where participants have to either match tasks and methods themselves or pairs of tasks and methods will be shown, and participants have to decide on the usefulness of this match. Visualization, psychology, and human-computer interaction experts will be used as participants for the experiment.

3.5 Creation and Evaluation of New Visual Analysis Methods

Once the matching process is defined, and experts have been interviewed about the visual analysis methods, missing visual analysis methods will be developed. A new visual analysis method will then be evaluated in a user experiment to investigate its usability, as well as how good tasks and goals intended can be solved.

3.6 Creation and Validation of the Analysis Model

The last step is to combine the parts of the experimental methods, the analysis tasks, and the visual analysis methods into an analysis model. This includes the definition of guidelines how the decision model uses the input data, and how it interacts with the user. The analysis model will first be developed based on an exemplary workflow for one type of empirical visualization experiment to show how the model operates. This workflow will be evaluated in a case study with an appropriate visualization domain. After showing that the model works for one visualization domain further domains are added and evaluated.

4 Related Work

In this section related work of analysis tasks, and visual analysis methods will be presented and discussed. Analysis tasks mainly focus on tasks in combination with the analysis of eye tracking experiments.

4.1 Analysis Tasks

Andrienko et al. [1] define analysis tasks for eye tracking data which they divide up into two major categories: "tasks focusing on areas of interest (AOIs)" and "tasks focusing on [eye] movements". These analysis tasks are collected in the following list and are solely defined for eye tracking experiments independent of the domain. They can be used as a starting point to define appropriate analysis tasks specifically for empirical visualization experiments. The defined analysis tasks are the following:

- Overall spatial pattern of movements; Relation to display content or structure;
- General character of movements; Individual spatial pattern of movements; Relation to display content or structure; Individual search strategy;
- Spatial pattern of attention distribution; Relation of attention foci to display content or structure; Repeated visits;
- Relation of movements to particular AOIs; Returns to previous points; Places where users have difficulties;
- Connections between AOIs; Presence and frequency of repeated moves;
- Comparison of trajectories;
- Comparison of spatial patterns of movements of different user groups;
- Comparison of spatial patterns of attention of different users or user groups;
- Comparison of spatial patterns of movements on different displays;
- Comparison of attention distribution on different displays;
- Evolution of eye movements over time; General search strategy; Types of activities and their temporal order;
- Changes of attention distribution over time;
- Frequent/typical sequences of attending AOIs; Cyclic scanning behavior.

4.2 Visual Analysis Methods

Visual analysis method from the eye tracking domain have been developed. The most prominent are heat map and scanpath visualizations which will be briefly presented in the following. Other visualizations techniques have been developed to meet different requirements of different application domains, such as for example the circular heat map transition diagram by Blascheck et al. [2], the time radar trees visualization tool by Burch et al. [3], transition matrices by Goldberg et al. [5], the parallel scan path visualization by Raschke et al. [8], or eyePatterns by Tsang et al. [10].

Heat Map: A heat map uses fixation data of multiple or all participants, sums it up, and visualizes it using a color scale. This visualization technique can be used to get an overview about eye movements of all participants and to define areas of interest [6].

Scanpath: A scanpath visualizes the fixation data of each participant individually. This visualization technique shows the complete eye movement path of one participant, or multiple participants by using different colors. Each fixation is indicated by a circle where the radius of this circle is based on fixation duration, saccades are visualized by lines connecting these fixation circles [6].

5 Application Example

To illustrate how our analysis model can be used in an empirical visualization experiment we will describe the evaluation a real eye tracking study from the information visualization domain.

We will use the study described by Burch et al. [4] who compared two visualization techniques, the "Time Line Trees" and the "Time Radar Trees". The participants had to answer warm-up, counting and correlation questions. The data used for the stimuli was related to soccer. Participants had to answer 18 questions, 16 with a clearly determined correct answer.

Collected data in this study was the eye tracking data for each participant. The participant data contained information about mathematical backgrounds, video gaming skills, and soccer interests, as well as gender and age. The completion times and answers were also collected during the experiment for each question.

Analyzing this experiment using our analysis model, and using the defined steps in section 2, the first step is to get the information about the experimental design. In this experiment the experimental design method is a true experiment with a between-subject design, as participants were split into two groups randomly, one for each visualization tool. The data collection method used eye tracking, and questionnaires to collect eye movement date, completion times, and accuracy rates. The visualization technique is a radial visualization type compared to a Cartesian representation. One possible research question in this experiment could be to find out why "Time Line Trees" can be used to answer counting questions faster than "Time Radar Trees". To examine this question the analysis task "Overall spatial pattern of [eye] movements; relations to display content or structure" from section 4.1 could be chosen. This analysis task would require a visual analysis method showing all eye movements of one participant, which could be a scan path visualization of each participant. Other potential visual analysis methods might be better suited to investigate this analysis task, like the parallel scan paths by Raschke et al. [8], this could be used in the second iteration of the analysis model.

6 Conclusion

In this paper we investigated if an analysis model can be created to support visualization designers in evaluating their empirical visualization experiments. We formulated how a potential analysis model could look like by introducing the concept of analysis tasks and visual analysis methods. The appropriate visual analysis methods and analysis tasks will be chosen according to the experimental design, and experiment conduction. In an application example we illustrated how the analysis model can be used to investigate an empirical visualization experiment. We further defined the appropriate methods and procedures that need to be used to realize our analysis model. The implementation of this model will be future work.

References

1. Andrienko, G., Andrienko, N., Burch, M. and Weiskopf, D.: Visual Analytics Methodology for Eye Movement Studies. Visualization and Computer Graphics, IEEE Transactions on, 18(12), 2889–2898 (2012).
2. Blascheck, T., Raschke, M. and Ertl, T.: Circular Heat Map Transition Diagram. In Proceedings of the 2013 conference on Eye Tracking South Africa (2013) (to appear).
3. Burch, M., Raschke, M. and Weiskopf, D.: Exploring Spatio-Temporal Data Modeled as Dynamic Weighted Relations. In Proceedings of Workshop on Visual and Spatial Cognition (2013) (to appear).
4. Burch, M., Bott, F., Beck, F. and Diehl, S.: Cartesian vs. Radial - A Comparative Evaluation of Two Visualization Tools. In Proceedings of 4th International Symposium on Visual Computing (ISVC08), Las Vegas, Nevada (2008).
5. Goldberg, J. H., Kotval, X. P.: Computer Interface Evaluation Using Eye Movements: Methods and Constructs. In International Journal of Industrial Ergonomics, 24, 631–645 (1999).
6. Holmqvist, K., Nyström, M., Anderssson, R., Dewhurst, R., Jarodzka, H., Van de Weijer, J.: Eye Tracking – A Comprehensive Guide to Methods and Measures. Oxford University Press (2011).
7. Keim, D. A., Kohlhammer, J., Mansmann, F., May, T. and Wanner, F.: Mastering The Information Age – Solving Problems with Visual Analytics. Eurographics Association (2010).
8. Raschke, M., Chen, X. and Ertl, T.: Parallel Scan-Path Visualization. In Proceedings of the 2012 Symposium on Eye-Tracking Research and Application, Volume 2012, 165–168 (2012).
9. Tory, M. and Möller, T.: Human Factors in Visualization Research. In IEEE Transactions on Visualization and Computer Graphics, Volume 10 (1), 72–84 (2004).
10. Tsang, H. Y., Tory, M. K. and Swindells, C.: eSeeTrack - Visualizing Sequential Fixation Patterns. Visualization and Computer Graphics, IEEE Transactions on 16(6), 953–962 (2010).