MASTER IN ARTIFICIAL INTELLIGENCE

MASTER THESIS

Grounding Semantics in Robots for Visual Question Answering

Author:
Björn WAHLE

Supervisor:
Prof. Dr. Luc STEELS
Universitat Pompeu Fabra (UPF)

Tutor:
Ulises CORTÉS GARCÍA
Universitat Politècnica de Catalunya (UPC)

Facultat d'Informàtica de Barcelona (FIB)
Universitat Politècnica de Catalunya (UPC) - Barcelona Tech

Facultat de Matemàtiques de Barcelona
Universitat de Barcelona (UB)

Escola Tècnica Superior d'Enginyeria
Universitat Rovira i Virgili (URV)

June 23, 2019

MASTER IN ARTIFICIAL INTELLIGENCE

Abstract

Grounding Semantics in Robots for Visual Question Answering

by Björn WAHLE

Since the rise of deep learning, image segmentation and especially object recognition and semantic segmentation are dominated by convolutional neural networks with remarkable performance compared to earlier approaches. However, deep learning is never entirely free of bias and needs large annotated datasets for training, moreover results are often surprisingly brittle and is biased on internal categories which do not correspond to human categories. An alternative to deep learning for the grounding of semantics is based on evolutionary linguistics. In this approach, E embodied agents (robots) play language games to learn and ground semantic concepts in an emergent, evolutionary way. To ground visual perceptions in language games a minimal bias vision system that is able to detect arbitrary objects and describe them in abstract ways is necessary. In this thesis I describe an operational implementation of an object detection and description system that incorporates in an end-to-end Visual Question Answering system and evaluated it on two visual question answering datasets for compositional language and elementary visual reasoning. The images of the datasets represent simple scenes containing basic geometric solids of different color, size, shape and material. In the first dataset, the CLEVR dataset [41], the images are synthetic generated images. The second dataset is contributed by this thesis and consists of images captured by the NAO robot.
The results show that the proposed system is able to detect objects in both datasets with a high precision and recall and outlines how the implemented object descriptors can be used to ground semantic attributes of the basic solid geometric solids.

Acknowledgements

The research in this thesis took place at Institute for Evolutionary Biology of the Universitat Pompeu Fabra (UPF) and the Consejo Superior de Investigaciones Científicas (CSIC). The project was partially funded by the CHIST-ERA project Atlantis.

This thesis would not have been possible without the help of a lot of people and the robust software they developed. Firstly, without the existence of the python library *OpenCV* [10] the implementation could have never done in the limited amount of time available. Being able to use a stable, high performance and extensible toolkit of state-of-the-art computer vision algorithms gave me the freedom to explore and experiment in a very efficient way. Secondly, the language framework Babel2 with its main components FCG and IRL, provides an amazing foundation for open-ended language experiments, suited perfectly for Visual Question Answering.

Next I want to thank the people that helped me to get started in this project and taught me much more than I could have learned myself or in the masters program: First I want to thank the team of the Artificial Intelligence Lab of the VUB Brussels to let me be part of this project. I want to especially thank Jens Nevens whose technical help with the Babel2 framework and whose general advice on my project were enormously helpful. I want to thank my supervisor Luc Steels, who introduced me to a new point of view of Artificial Intelligence and was a great advisor and motivator that always made me believe in my work.

Contents

Chapter 1

Introduction

On the way to general artificial intelligence, visual question answering is one of the most important fields of research. While already great progress has been made on datasets like the CLEVR[41] or VQA[3], most studies take place in a simulated, static environment. Real world applications of VQA will need to face a variety of circumstances, including varying lighting conditions, limited vision and computing capabilities as well as incomplete information due to noise in perception or comprehension and human friendly interaction. When executing VQA in robots, the introduced complexity by these circumstances can be faced by using the possibilities of autonomous behaviours of the robot and special ways of interaction to recover from failure. Contrary to the current state-of-the-art deep learning methods for object detection and visual reasoning, the system developed in this project is not trained on a large training data set, but is intended to build the basis for a incremental evolutionary system, that learns by emergent communication.

An example for a visual question answering task we are working in this project would be: Given the question: *What color is the big cube to your right?* and an environment of geometric solids in the robots vision, the system first analyzes the scene by segmenting it into objects, then comprehends the question and constructs a functional program and finally links the semantic categories (*big*, *cube*, *to your right*) to the perceived objects and answers the question by saying *red*, which is the semantic label for the detected object which is referred to in the question.

This thesis will explore the challenges of grounding semantics for visual question answering in robots in an environment with primitive objects in basic shapes, questions in the style of the CLEVR dataset[41] and the NAO robot as the visual question answering agent. The thesis is organized in 8 chapters:

Introduction In the first part I will discuss the basic concepts of VQA before I will explain the motivation of this thesis and give an overview of its objectives. Furthermore I will explore the main challenges being faced and summarize the related work in this field.

Language Understanding In chapter 2 I will give an overview over the necessary structures for a continuously adapting and improving linguistic system of the robot. It will give the theoretic foundation for the language processing system used.

Scene Understanding Chapter 3 will introduce the necessary concepts of image understanding by discussing main methods of image processing, image segmentation, and object description.

Contributions Then in chapter 4 I will discuss the contributions of this thesis to visual question answering and how to ground semantics without machine learning.

VQA System In this chapter the VQA system used in the experiments will be discussed. I will give an overview over all parts and a detailed explanation of the scene understanding algorithm.

Experiments After having discussed the theoretic part of the thesis, I will describe the executed experiments and how to evaluate them in order to show the value of the contributions of this work. Then I will discuss the obtained results.

Conclusions and Future Work In the last chapter I will sum up my conclusions of the work and highlight the value of the contributions before I finish by explaining future work opportunities that arose from this thesis.

1.1 Visual Question Answering

Visual Question Answering (VQA) [3] is the multi-discipline problem of answering open-ended and free-form natural language questions given an image containing the information to answer them. Its three disciplines that need to be solved in order to develop a VQA system are Natural Languange Processing (NLP), Computer Vision (CV) and Knowledge Representation and Reasoning (KR). It is believed that solving this problem and its applicable task would lead to general artificial intelligence and is therefore seen as an "AI-complete" problem. On the other hand, the experiments in this work will be performed in a less open-ended and free-form environment then classical VQA problems. In order to show the foundation of VQA in robots the environment will be limited to an indoor environment with a few basic objects and the questions will be from a specified set of types.

1.2 Related work

In the fields that are discussed in this thesis, much research has already been done. I will summarize the main approaches for recent work in VQA, image segmentation and the grounding of semantics in embodied agents (robots).

1.2.1 Visual question answering

Although visual reasoning and natural language understanding have been studied for many years before [5, 86, 8] and great advances in the field of image captioning and visual description [56, 43, 91, 27] as well as in text-based question answering

[26, 25, 90], the task of visual question answering as such was proposed only recently in [3]. Since then, various combinations of convolutional neural networks (CNNs) and recurrent neural networks (RNNs) of different architectures were able to achieve good results [31, 39, 58, 51, 44]. Furthermore, several extensions to VQA have been proposed: Embodied Question Answering (EQA) [20] extends the task by embodying the visual reasoning system as an agent that needs to explore the scene to answer the questions, Interactive Question Answering [33] extends this approach by adding the necessity to interact with the environment. Recently, GQA [40] was introduced as an extension to VQA, providing more realistic questions and compositional question answering. The CLEVR dataset aiming to reduce bias by randomly generating scenes for compositional question answering was introduced in [41]. As seen in [45], deep learning methods have significant problems with the *same-as* task, where objects are tested for equality on visual attributes.

However, most of the approaches to solve VQA and its extension make use of deep learning with large training sets. The performance of the agent in answering questions is therefore dependent on the quality of the dataset and is not incremental, as the agent does not continue learning and adjusting after the training phase. In [21], two agents are cooperatively playing an image guessing game, where one agents randomly picks an image of a set of images and the other agent has to guess which image was picked in a dialogue way and adjusting their systems by reinforcement learning. In [47], reinforcement learning is used to learn visual relationships and semantic attributes of objects.

Furthermore, recent studies found that many of the recent VQA models are heavily driven by superficial correlations in the training data and lack sufficient visual grounding. For example, many question can be answered with a high accuracy without analyzing the image, as for example existence questions in the VQA dataset starting with "Do you see a ..." testing for the existence of a specific object in the image can be answered blindly with "yes" achieving a accuracy of 87% [35]. To balance the VQA dataset and encourage the necessity of a robust image understanding system, different splits and additional questions have been proposed in [35, 93]. Since then, also models that intend to overcome strong priors from the training data and providing more visual explainability have been proposed, as for example in [2].

1.2.2 Image segmentation

In image segmentation and especially object detected, great advances have been achieved since the rise of convolutional neural networks. State-of-the-art object detectors like YOLO [64, 63] or Mask-R-CNN [37] provide outstanding results on object detection challenges like the CoCo [48] or PASCAL VOC [24] but also criticism on deep neural networks (DNNs) has been formulated in for example [57] where the authors present examples of images that a human would never label as certain object, but a DNN does with high (99%) certainty.

The research in traditional image segmentation methods which do not employ DNNs has decreased since the rise of the rise of those. For many application-dependent tasks, the use of traditional image segmentation is still necessary. The latest advances in image segmentation are for example [4], in which the authors propose a new contour detector and a image segmentation method which outperforms state-of-the-art methods.

Lastly I want to comment on the state-of-the-art on the CLEVR dataset[41]. While many end-to-end implementations achieve answering accuracy over 90%[39], [70], [42], [61], there are few approaches without machine learning in either the language, reasoning or perception part of the VQA task. To my best knowledge, there are no approaches to the CLEVR dataset that do not use a CNN for object detection.

1.2.3 Grounding semantics in robots

The challenge of grounding semantics (incrementally) in robots has been under active research in the last years as well. One of the first projects attempting to ground semantics by emergent incremental learning was the Talking Heads experiment [80], being followed by experiments in mobile agents in cultural language evolution [76]. In [54], the authors propose a Grounded Situation Model (GSM) as a unified model for perception and semantics in robots. In [34] and [72] the creation of semantic constructions to express visual relations is studied.

1.3 Emergent incremental learning

A different approach to the use of deep learning in order to achieve general intelligence is to learn in an emergent way in a multi-agent environment by communication and focusing on the evolution of meaning adapted to the needs to achieve a predefined goal. This is usually set up as language games, in which a pair of agents with initially no common language or concept understanding have to emerge concepts by iterative guessing or naming games. Each of the agents analyzes its environment and describes it internally with abstract descriptors and then one agent, the speaker, creates or uses an already existing concept based on the most discriminating descriptors to describe an object that it wants to draw the attention too and then produces an utterance. The other agent, the listener, tries to comprehend this utterance and guesses the object that the other agent chose. If it is not possible to comprehend the utterance of the speaker, the listener asks the the speaker to solve. Based on the outcome of the game, both agents adapt their internal concepts to increase the communicative success.

1.4 Motivation

In recent years, overwhelming advances have been made in the field of Artificial Intelligence, especially in the sub fields of Natural Language Understanding (NLP) and Computer Vision (CV). In the frame of the Robot Soccer Cup, efficient image segmentation algorithms are used with advanced robot movement technologies to teach robots to play football. The solutions for in this application are very tailor made and adapted to solve one problem instead of providing solutions in a more general framework. In other works, the field of embodied visual question answering was faced by agent simulations in a 3D environment. However, most of the latest research relies heavily on the use of deep neural networks and the presence of large training data sets. These deep learning approaches usually do not provide a sufficient explanation for their results. In the state-of-the-art object detection algorithms

[63, 37], usually the bounding boxes with a vector of class probabilities that the detected object belongs to is returned, but the explanation of why it was classified like this can only be extracted by deeply investigating the neural networks layer activations and the training data. In this work I want to tackle this issue by developing an image segmentation method that returns numerical object descriptors which can be used to ground semantics in communication.

Especially I am motivated to fit the methodology developed to the framework of the earlier explained emergent evolutionary learning. The usage of large training data sets to learn concepts is contradictory to the idea of emergent learning by communication through language. Therefore I am using traditional, cognitive computer vision algorithms and describe concepts in a numerical, abstract way, which can be used to learn concepts and to understand the environment in a multi agent setup.

Second it aims to implement an image segmentation method that is not depending on a large training data set and deep learning techniques and provides results with a high level of explainability that can be used in the frame of emergent evolutionary learning and other similar applications.

1.5 Objectives

After discussing the motivation for my research and development, I want to state the objectives of this thesis. Having explaining my doubts about the use of deep learning in order to ground semantics and discussed the approach of incremental evolutionary learning, my objectives in this thesis are two-fold: First I want to implement a image segmentation algorithm that is unbiased by training data that can be used to detect arbitrary objects in images without the semantic knowledge of what the object is. The algorithm should be deterministic and provide an appropriate base to describe the detected objects. The second objective of the thesis is to implement a set of object descriptors for the detected objects that can be used to ground semantics to the detected objects. The main idea of this objective is not to find the optimal grounding, but to discuss and evaluate existing descriptors and implement a set of object descriptors.

1.6 Challenges

After having revised related research, explaining my motivation and my objectives in this thesis, the challenges to face in order to achieve the objectives will be shortly explained.

Incomplete information due to noise As we are working with the robot, we are mostly using sensory data: The microphone of the robot, the camera of the robot, touch sensors or the joint rotations of its limbs. Especially in the vision system, noise is introduced almost all the time. From low level noise that comes from the poor camera quality to visual artifacts introduced by shadows or dirt, the different types of noise often lead to incomplete information on the scene. The presence of noise in most of the sensory data is one of the main challenges to face when solving artificial intelligence problems with robots. In the scope of the end-to-end VQA answering

task, also the output of one of the systems can be noisy: The image segmentation might return incorrect detections or descriptions or the comprehension of the question might be wrong. The correct answering of a question in respect to the noise that is possibly introduced in any of the systems stages is one of the biggest challenges researchers are facing.

Minimal bias object detection Grounding semantic models with vision requires an abstract and robust image segmentation and object recognition system. State-of-the-art image analysis systems rely deep learning approaches which use huge sets of annotated training data. In the frame of this work, the usage of annotated training data would bias the autonomy and emergency of the robot in his behaviour and is therefore not an option. From emergent learned colors, shapes and spatial attributes an abstract and minimal set of basic concepts needs to be formed that allows the evolution of a more complex grounding ability.

Further challenges will be explained in the image understanding chapter.

Chapter 2

Language Understanding

Of the two-folded problem of Visual Question Answering (VQA), naturally in its applications the Language Understanding part, the process of comprehending the question is the first necessary step in order to produce a semantically correct answer. In this chapter the basic concepts of language understanding as we will use it in this thesis will be examined.

2.1 Construction grammars

Construction grammars are a cognitive approach [23] to express linguistic knowledge. One of the main characteristics of cognitive linguistics is that language always needs to involve meaning. Therefore, the fundamental organizational units in language processing are pairings of form and meaning. Every unit in a construction grammar holds semantic and syntactic information.

2.2 Evolutionary linguistics

A key aspect on the way to modeling natural language in (mobile) agents, is the fact that language is not static, but evolving. One may be able to implement a language system for agents that is able to understand and communicate well on a specific topic. Even a broader set of tasks could be implemented in a more or less flexible way, but a closed grammar will never be able to parse and comprehend all semantics that humans understand. Therefore an approach that models the evolution of language, both in an individual as in cultures or populations, is favorable. The linguistic theory called *the selectionist theory of language evolution* [77] aim to model this by two basic concepts: *selection* and *self-organization*. Linguistic selection or selectionism splits the language evolution process in two parts: *Generating* possible variants by small modifications of existing concepts and *testing* whether the generated variants fit desired selection criteria like *communicative success*. These concepts were first proposed by Darwin. The other concept, *self-organization* describes the collective process of aligning ones internal concepts based on communicative interactions. For example, when a misunderstanding between two persons because of the misalignment of internal concepts occurs, both the speaker and the hearer will adjust their internal concepts autonomously. Imagine a situation in which the speaker asks the hearer for an object and describes it by its color: The question *Can you give me the turquoise ball?* might be understood wrong in a context with many green and blue balls. If the

hearer fails to identify the ball that the speaker referred to, he needs to ask for clarification. The speaker then points to the ball he referred to. After this unsuccessful communicative interaction, the speaker knows that his internal concept of the color *turquoise* might not be aligned optimally with the generally known concept of the color and therefore needs to align it. The hearer on the other side has either learned a new concept and can add it to his inventory or adjust the conflicting concepts that led to the misunderstanding.

2.2.1 Grounding semantics

A further important issue is that of grounding, since language is grounded in experience. Humans understand many basic words in terms of associations with sensory-motor experiences or interactions. Basic words like "red", "light" or "in front of" can only be understood by humans if they interact physically with their environment. Grounding in agents is usually conceptualized in three steps:

1. **Invention**: if a perceived thing can not be explained by using one of the concepts in their inventory, it is necessary to invent a new prototype that links the perceiving attributes (optimally in the most discriminating way) to a newly invented semantic category and lexical item.
2. **Adoption**: another way of extending an agents internal inventory is to adopt a concept in interaction with another agent. To adopt a concept, an agent needs to invent a new prototype, link it to a semantic category and link it to the adopted lexical item.
3. **Adjustment**: if an agent experiences by interaction, that another agent has a different lexical item for the same perceived thing, it can adjust its internal links.

2.2.2 Language games

A method to simulate the way how humans supposedly learn and acquire language is by playing language games in a population of embodied agents. A language game is played by two agents at a time. One agent is the *speaker*, the other one is the *hearer*. The speaker has to produce an utterance, which the hearer attempts to comprehend and act accordingly. The speaker then signalizes if the executed action matched the intent of his utterance. Then one of the agents or both agents align their internal concepts based on the communicative success in the game: If action and intent match, the game was successful, otherwise it was not. Played in a population of agents, in each language game, two agents are randomly drawn from the population of agents. Like this it can be evaluated if and after how many games the internal concepts of the agents in the population align to a certain degree. In general, a wide range of different language games can be designed. A simple game is the *color naming game*. In the color naming game the goal is to link color names to the internal color concepts. Both agents perceive a shared environment with three different colored objects. The speaker then selects one of the objects and uses a an utterance describing the object by its color in the most discriminative way. If the agent does not yet have a word for the color, he invents one and links it to its internal color concept. Then the hearer comprehends the utterance and attempts to identify the referred object by classifying the perceived objects in color categories and linking them to his lexical inventory. If

the lexical utterance of the speaker is unknown to the hearer, he has to ask for the solution, otherwise he points to the object that he assumes it the object that the speaker referred to. The game is successful if the hearer points to the object that the speaker is referring to.

2.3 Fluid construction grammar

Fluid Construction Grammar (FCG) [75] is a linguistic formalism for a defining the inventory of lexical and grammatical conventions that language understanding requires. It was designed for open-ended, grounded dialogue and is therefore suited well for visual question answering in robots. A computational formalism is necessarily based on a particular perspective on language. For FCG, this perspective is inspired by research in cognitive linguistics in general and construction grammar in particular. FCG does not have a clear structure, leaving the developers a high degree of freedom in defining their language grammar to be able to represent syntactic and semantic language formulations. One of the main building blocks of FCG are constructions. As constructions are a core concept in linguistic theory in many approaches, it is worth defining the notion of a construction in FCG. In FCG, a construction is a regular pattern of usage in language. It can be a single word, a combination of words, an idiom or a syntactic pattern with a conventionalized meaning and function. A construction can both contain semantic and syntactic information. Meaning and functional parts of a construction are stored conceptually in a *semantic pole*, whereas aspects related to form as syntax, phonology, morphology and phonetics are in a *syntactic pole*. The different types of constructions can be defined by the grammar developer. In general, they can be classified on a continuous scale between simple constructions containing a single item and of lexical nature and abstract grammatical constructions which for example describe relations between constructions on a complex level.

2.3.1 Transient structures

Both parsing and production in are done with the some constructions in FCG. The computational task to read a sentence and parse it to understand its meaning as well as the reverse process of producing an utterance that represents a given meaning are very complex, being far from straight forward iterative application of rules. Instead, it can be seen more as a problem solving task in a large search space. Therefore, intermediary structures that represent the structure of the constructions in a given state of the production or parsing process are important. In FCG, these structures are called *transient structures*. They consist of a a set of units, which an hold any relevant language features as for example semantic, syntactic or morphological attributes. A simple example would be a transient structure for the construction "the dog", which would contain the lexical units "the" and "dog", as well as the nominal phrase consisting of the sequence of the lexical units. Each of the units can hold further attributes that are useful for production and parsing as for example the part of speech, if its plural or singular, but also semantic information as for example "dog" belonging to the class of animals. Constructions now can be applied to each of the unit to either transitions to a more abstract level of the transient structure, or to a more defined. In general, for both parsing and production, the path from the initial

transient structure, consisting either of the raw lexical words in its order, or in the most abstract meaning abstraction, to the final structure can be thought of as a chain with one construction applied to each unit per step:

$$T_{init} \longrightarrow T_1 \longrightarrow T_2 ... \longrightarrow T_{final}$$

To detect which construction can be applied to a unit of the transient structure, a *matching step* is executed to test whether a construction is compatible with the current transient structure. Then a *merging step* is applied which extends the involved units with the new information.

2.3.2 Basic principles

The basic principles of FCG will be covered in this section. Firstly, FCG aims to represent constructions in a declarative way. That basically means that transient structures have the same form as constructions, making it easy to learn and implement. Furthermore, all types of constructions are build without formal differences. This is of advantage in the conceptual scalability of a grammar, as at any time new constructions of a more abstract level as well as of a more lexical level can be added without changing the overall architecture. In addition, constructions in FCG follow the reversibility principle. This ensures that each successful parsing also always can be successfully used for production and vice-versa. This does not necessarily imply that the producing an utterance with the final transient structure of the parsing of a sentence will result in the exact same sentence. To follow this concept, constructions in FCG are defined in a bi-directional way: The semantic pole captures meaning, while the syntactic pole captures aspects of form. In language production, constructions trigger based on their semantic pole and add information contained in the syntactic pole. In parsing, they trigger based on the syntactic pole and add information contained in the semantic pole.

2.3.3 Technical implementation

FCG is fully implemented in system call the *FCG-system* in Lisp and has support for all common Lisp dialects. Its core component, the *FCG-interpreter*, is used to perform basic operations, such as tools for parsing and production. Furthermore it contains the *FCG-monitor* which is used to monitor the steps of parsing and production and observing success rates and other statistics when performing test cases on a certain grammar.

2.4 Incremental Recruitment language

Incremental Recruitment Language (IRL) [78, 79] is a general system for representing the procedural semantics of utterances. IRL can be used to combine different cognitive operations into complex conceptualization strategies. A cognitive operation for example is to compare two objects by their color. Technically, IRL can be seen as a functional program that can be run on a context of cognitive objects. The functional program is structured as a semantic network, where each node is a

semantic function with a number of arguments. These arguments are specified using variables, denoted with a starting question mark `?` as for example in `?color`. Nodes in the network are connect when their functions share the same variable. To be able to evaluate a IRL program, these variables need to be bound to specific values. For example, the variable `?color` can be bound to a particular color value `red` by using the bind operation: `(bind color-category ?color red)` where `color-category` defines the type of the variable. In comparison to functions as we know them in mathematics or programming, these semantic functions do not have a particular return value. Instead, the execution of a semantic function invokes finding possible values or a possible value for the unbound arguments. For example, a semantic function that was implemented to filter a set of objects by their shape called `(filter-shape ?filtered-set ?source-set ?shape)` could be called either with `?source-set` and `?shape` bound to specific values, and the variable `?filtered-set` would be computed by the program, or it could be called with `?filtered-set` and `?source-set` bound to values and the program would compute the value for `?shape`. Furthermore, if wanted, the program could even be invoked just with `?source-set` bound to a value, then the program would compute the possible combinations of values for `?shape` and their corresponding values for `?filtered-set`. Depending on the context, the ontology and the implementation of the program, it might be possible that the number of possible solutions is infinite. In this case, the developer needs to implement procedures to detect and prevent such cases. The evaluation of a semantic IRL network therefore results in finding possible values for all unbound variables in the network. A common practice to validate if only one solution for a variable of a semantic network has been found is therefore to check for uniqueness by implementing a semantic function `(unique ?target ?value)` that test if `?value` only consists of one unique value.

2.4.1 Technical implementation

As well as FCG, also IRL was implemented in Lisp and supports all common dialects. Similar to the FCG, it is a lightweight framework that only defines a couple of atomic building blocks as for example the entity, which is the parent class for everything that can be bound to a slot. Additionally it provides primitive functions for binding and more complex functions that for example check a IRL network for syntactical validity.

Chapter 3

Image Understanding

This chapter highlights the role of image understanding when grounding semantics in VQA. The objective of processing images is to obtain an unbiased and noiseless representation of the local scene. Foreground, background and other basic concepts are the atomic building blocks of the world representation. Detected objects are described by basic numerical features. In order to obtain a stable semantic representation it is important that these common atomic concepts are well defined and contain as less semantic bias as possible. In the first part of this chapter I will explain what kind of preprocessing is necessary. Then in second part the different concepts of traditional image segmentation will be covered. In the third part I will discuss how the detected objects can be described to be further used in the VQA system.. Lastly I will summarize the main challenges in image understanding for the system developed in this thesis.

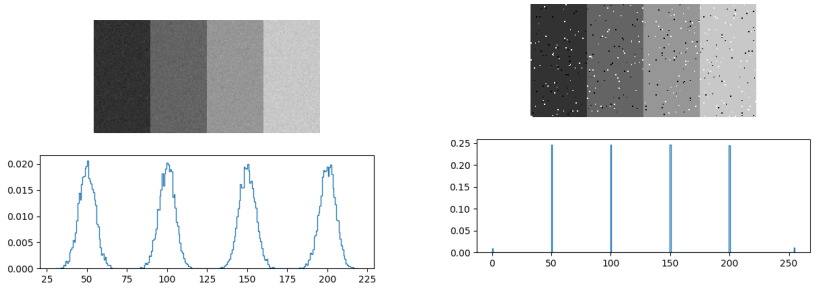
3.1 Preprocessing

For most computer vision applications, the raw image data contains too much detail or is not in the appropriate format for the task. Therefore, a set of preliminary transformations can be applied to the raw image data to obtain a pre-processed image that contains less information which is not needed for the task and enhances the information that is useful for the application. The most common preprocessing step is to remove noise from the image. The term noise in computer vision refers to artifacts in the image that do not have any high level semantic meaning for the world representation. Noise is usually introduced by bad lighting conditions, dirt on the camera lens or dirt on surfaces, but also by errors in the quantization process. A first step in image processing therefore is the removal of noise in order to obtain a clean image.

Additionally to ease the sequential CV tasks, it is a common practice to reduce the color space and enhance edges.

3.1.1 Noise models

To understand which techniques can be helpful when denoising images, it is important to know which types of noise to expect in the captured images. While more noise models exist, I will focus on discussing the main models that we will have to cope with in the camera images of the robot.



(A) Gaussian noise following a normal distribution where $\mu = 0, \sigma = 5$

(B) Salt and Pepper noise with a probability of 0.02 and equal amount of black and white noise pixels

FIGURE 3.1: Noise examples for a image with 4 gray levels and their corresponding gray level histograms

Gaussian noise is statistical noise having a normal distribution as its probability density function which is called the Gaussian distribution. The noise values are Gaussian-distributed by the function

$$p_G(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(z-\mu)^2}{2\sigma^2}} \tag{3.1}$$

where z is the gray level, μ the mean value and σ the standard deviation. Gaussian noise is often aroused when acquiring the digital image data because of poor lighting conditions but also because of high temperature in the camera sensor or electronic circuit noise. [13]

Salt and Pepper noise is a type of noise represented by sparsely occurring pixels being either black or white. These occur independently of the intensity of the original pixel or its neighborhood. It does usually occur because of bit-errors in transmission.

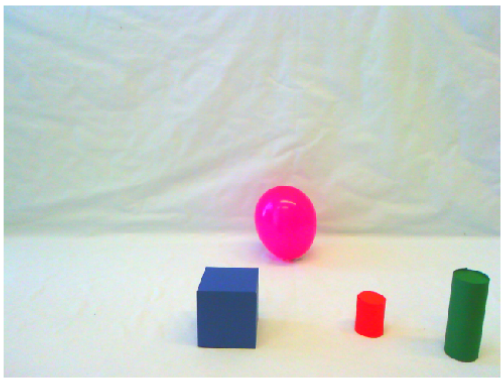
3.1.2 Pre-processing techniques

Smoothing

One preprocessing method often used on digital images is smoothing. Under the assumption that most noise in the image is Gaussian, it reduces noise by blurring the image. Most hard edges are turned into gradients, artifacts in mostly uniform regions are eliminated.

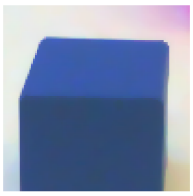
Gaussian smoothing Gaussian smoothing or Gaussian blur is a noise removal technique that relies on the assumption that noise is introduced following a normal distribution. Gaussian smoothing is applied to an image by convolving the image with a Gaussian function. For a 2D image the mathematical function for the convolution is defined as following:

$$G(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}}. \tag{3.2}$$

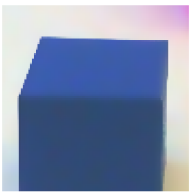


(A) Original

(B) Gaussian



(C) Median



(D) Bilateral

FIGURE 3.2: Comparison of different smoothing methods. The upper image shows the original unprocessed image. Images 3.2b to 3.2d show a zoomed-in part of filtered images with a Gaussian filter, Median filter and a Bilateral filter with a (7,7) pixel window.

The effect of Gaussian smoothing is the reduction of noise and detail. The resulting image looks blurry. A disadvantage to this method is that edges are not preserved, thus this noise removal technique is not useful if edge detection methods are applied afterwards.

Median filtering Median filtering is a non-linear approach to reduce noise by blurring. For each pixel in the image, it calculates channel-wise the median value of the neighbor pixels. The more neighbors around the pixel are used for the median calculation, the stronger is the effect of the noise reduction. Similar to Gaussian smoothing it also returns a more blurry version of the image but is preserving edges to a certain extent. It is often used to reduce detail.

Bilateral filtering To overcome the poor edge preservation abilities of Gaussian smoothing and Median filtering, in [84] a non-linear filtering approach was introduced. Similarly as in the other presented methods, the intensity values of nearby pixels are used to calculated the filtered pixel intensity, but in bilateral filtering the spatial and domain filtering are combined and used to weight each nearby pixel in the calculation. Furthermore it can be applied to all color channels at once.

Denoising

In comparison to smoothing, denoising methods attempt to remove noise from the image without changing the underlying structure at all. Edges and all level of detail should be preserved.

Total variation minimization This method is a constrained optimization type of numerical algorithm that minimizes the variation of the image subject to constraints based on the statistics of the noise. [67] The general optimization problem to be solved is

$$\min_y [E(x,y) + \lambda V(y)], \tag{3.3}$$

where x is the raw (noisy) image and y the denoised image to solve for. $E(x,y)$ is the L2-norm. $V(y)$ describes the total variance of the image and is defined as

$$V(y) = \sum_{i,j} \sqrt{|y_{i+1,j} - y_{i,j}|^2 + |y_{i,j+1} - y_{i,j}|^2} \tag{3.4}$$

for 2D signals like images. The goal therefore is to find an image y that jointly minimizes the total variance $V(y)$ and is as similar as possible to the original image. Since this optimization is not trivial and the originally proposed method of solving it was computationally expensive, in [15] a fast and converging algorithm was proposed and is since then the commonly used.

Non-Local Means Denoising Another noise reduction pre-processing method is Non-Local Means Denoising (NL-Means) [11]. Unlike the previously discussed methods, the filtered intensity values are not computed based on neighboring pixels, but on all pixels in the image, weighted by how similar the pixels are. The denoised value for a pixel i in the image I can be calculated by

$$NL[v](i) = \sum_{j \epsilon I} w(i,j) v(j), \tag{3.5}$$

where w(i,j) is the weight factor which depends on the similarity between the pixels i and j. This similarity is calculated by the similarity of the pixels neighborhoods of i and j and is calculated by

$$w(i,j) = \frac{1}{Z(i)} e^{-\frac{||v(N_i) - v(N_j)||^2_{2,a}}{h^2}}, \tag{3.6}$$

where $Z(i)$ is the normalizing factor:

$$Z(i) = \sum_{j \epsilon I} e^{-\frac{||v(N_i) - v(N_j)||^2_{2,a}}{h^2}} \tag{3.7}$$

and h^2 a parameter to control the degree of filtering.

Block-Matching and 3D filtering Another transform domain approach to image denoising is Block-Matching and 3D filtering (BM3D) [19]. This method also uses

non-local image information by separating the image into blocks. Then, by using a block-matching method, similar regions are found for each of the blocks of the image and are stacked in a 3D array. By using decorrelating unitary transforms on on these arrays a estimate for each block can be calculated. The final estimate is computed as weighed average of all overlapping block estimates. This method is patented and not available for free usage.

Feature enhancement

Since preprocessing is normally used as a primary step before applying other image processing techniques, a common goal of the preprocessing is to enhance the features needed for further methods. These might transform the image in a way that it does not realistic anymore, but improve the performance of image segmentation, object detection or other methods.

Mean shift filtering Mean shift filtering [18] is a filtering method based on the mean shift algorithm [32] to reduce noise, color space size and enhance edges. In contrast to Non-local means denoising and BM3D, mean shift filtering does originally not aim to preserve all structure and detail of the original image. The principle is to initialize a mean shift vector for each pixel of the image and iterate the mean shift towards the local image point with the highest density until convergence for each pixel. The new pixel value at the initialized location will be assigned the intensity value of its convergence point.

3.1.3 Contour detection

A further preprocessing procedure that is useful to understand the contents of an image is contour detection. A contour is the boundary of a region and is often visually seen by an abrupt change in intensity.

Laplacian of Gaussian The Laplacian is a 2D isotropic measure of the second spatial derivative of an image. It highlights rapid intensity changes and is therefore suited for edge detection. As it is highly sensitive to noise, the common method first introduced in [53] is to calculate the Laplacian of a Gaussian filtered version of the image. To combine these operations, the Laplacian can be applied to the Gaussian function. Equation 3.8 is used to calculate the filter kernel for the convolution with the image. To be used in digital image processing, a discrete version has to be calculated.

$$g(x,y) = -\frac{1}{\pi\sigma^4} e^{-\frac{x^2+y^2}{2\sigma^2}} \left(1 - \frac{x^2+y^2}{2\sigma^2}\right) \tag{3.8}$$

Canny Edge Detector The Canny Edge detector [12] is a multi-stage edge detector that relies on the intensity gradients of the image. First, a Gaussian filter is applied to reduce noise. Then the intensity gradients of the image are extracted with an edge detection operator like Sobel. Then non-maximum suppression is applied to eliminate double edges and only remain with the strongest local edges. Then edge detection operator is applied again but with a higher threshold, finding only the

(A) Raw image

(B) Total variation

(C) NL Means Denoising

(D) Mean shift filtering

FIGURE 3.3: Comparison of different denoising methods. The upper image shows the original unprocessed image. Below from left to right: The denoised image with total variation minimization, with non-local means denoising and mean shift filtering. The strong noise on the purple object is eliminated on all three images. The image filtered with total variation looks slightly blurry. The NL means denoised image removed big parts of the noise. Mean shift filtering removed well the noise on the purple object and enhanced edges very well, but the self shadow of the red object still contains noise.

strongest edges in the image. As a last step, all edges that are not connected to a strong edge are removed.

3.2 Image segmentation

In image segmentation we understand the task of segmenting an image into multiple segments that form the image. Since there are different detail levels of image segmentation, each applicable to different use cases, I will give an overview of the different types of image segmentation and then focus on the case that we are facing in this thesis.

Generally speaking, image segmentation is partitioning an image into regions of pixels that belong together by a specific criteria like spatial or color similarity. The level of detail of image segmentation depends on the application: As a pre-processing step or for artistic transformations of the image, the image might be segmented to a reduced color space, while for other applications it is desired to segment the image in background and foreground and each of the foreground segments represents one object. The latter case is also described as object detection. The goal is to find all pixels that belong together to form an object.

(A) Laplacian of Gaussian (B) Canny

FIGURE 3.4: Comparison of Laplacian of Gaussian (LoG) and Canny edges. The image has been smoothed with a Gaussian filter of size 3 before the edge detectors have been applied. Note that the LoG edges are not linked together and remain noisy, while Canny edges are crisp but still contain some incomplete edges.

Since there is no clear and well defined explanation of what is an object, the level of detail in image segmentation is always subjective and dependent of the application. For example both a car as also one tire of the car can be seen as individual object. It depends on the application to define the granularity of the segmented regions. Concluding from this, it is not possible to develop an image segmentation technique that works without any priors on any given image. A certain context knowledge is always necessary.

Object detection The task of object detection is to find regions that form objects in the image. While in theory, everything in a captured image can be seen as an object, as for example small details far away from the camera, usually object detection aims to find objects of interest in the foreground of the image. Therefore object detection is normally executed with context knowledge, identifying what the objects of interest are. A common task for example is face detection, where the image is filtered for human faces. A more general object detection without prior knowledge of the objects that need to be detected can be executed only on the foreground of the scene. The foreground of the scene describes the nearest and most prominent areas in the image.

Semantic segmentation Semantic segmentation describes the task of segmenting the image into regions that belong semantically together. Technically, each pixel of the image is assigned one or more semantic labels. This also requires the segmentation method to understand the semantics of the parts of the image. This task is hardly possible to solve without large annotated training datasets that are used to learn the semantics of the objects or regions in the images.

Instance segmentation While semantic segmentation only labels each region what semantic class it belongs to, it does not detect instances of a class as an object. Therefore a further step, segmenting the regions of a class into its instances, is necessary in some applications. Especially for regions with multiple occluding instances of the same class, this task is considered very difficult. This is the type of image segmentation that we are dealing with in this thesis.

FIGURE 3.5: Binarized image, foreground pixels are displayed in yellow, background pixels in purple.

3.2.1 Foreground detection

In order to obtain the foreground objects it is necessary to perform a binary classification on the pixels of the image in to background and foreground. Once the background pixels are identified, we can apply a mask to focus on the foreground objects as seen in figure 3.5. The foreground is defined as the objects or regions which are close to the view point, prominent and optionally dynamic, while the background are the static regions which usually are located at a further depth level. There are different approaches to separate background and foreground which I will shortly explain. Furthermore I will explain the most common image processing techniques on binary image.

Multiple images

In dynamic environments, background is usually defined as the regions of the image that are not moving while the foreground are the regions that contain the objects that move and change. Using a set of images background model can be found using statistical, neuro-inspired, fuzzy or other various techniques. This background model can then be used to be subtracted from the image to obtain the foreground regions. In the following I will explain the two main background initialization approaches where a set of images is used.

Background initialization in videos When a sequence of images, also called video frames, is given, a common procedure to obtain a robust background model is to use statistical models of each pixel in the subsequent frames. A basic approach for example is to calculate the temporal median image of the last n frames [49]. A more complex approach is fitting a Mixture of Gaussians to the background model as in

[60]. This approach to background-foreground classification is only useful under the assumption that the foreground objects are moving and the background is stationary. The most common use case is video surveillance.

Background calibration Another approach to detect foreground regions is to firstly calibrate a background model of the scene without any foreground objects by gathering statistical data on multiple images of the background as seen in [74]. Then the foreground regions can then be detected by a pixel-wise classification: If the distance of the pixels intensity to the corresponding mean intensity of the calibrated background is greater than the standard deviation, the pixel is detected as foreground. Outliers can be eliminated by morphological operations like closing. It should be noted that this method is not applicable if it is not possible to calibrate the background model by removing all objects.

Single images

If only a single image is available, different strategies for separating background and foreground are necessary. Many interactive approaches that require user interaction have been proposed [68], [16], [88]. In all of the proposed methods, a certain amount of context information needs to be available. If it is for example known, that the background is a wall or the sky, simple color segmentation methods can be used to extract the background. In other cases, transforms of the image like edge detection are used to find closed contours which are assumed to be part of the foreground.

Binarization

The term Binarization is used in image processing for the process of separating all pixels in an image into two groups and display it as a black and white image. Usually the group of black pixels describes the background of the scene, while the white pixels describe the foreground.

Morphological operations on binary images When processing binary images, a set of image transformation operations, called morphological operations (see for example [73]), can be applied. These morphological operations filter the binary image with a structuring element which is a square matrix consisting of 1s and 0s. The fundamental operations are erosion and dilation. In erosion, only pixels where all 1s in the structuring element are also 1s in the window that is tested, are set to 1. In dilation, all pixels where at least one 1 of the structuring element is matching, are set to 1. Erosion shrinks regions and eliminates small details, while dilation enlarges regions and enhances small details. Many morphological operations are compound operations, combining the fundamental operations and set-theoretic operations like union and intersection. The most commonly used are opening and closing. Opening is to first apply erosion, then dilation to the binary image, while closing first applies dilation and then erosion. The opening is used to separate regions that only have thin connecting bridges and eliminates small details, while closing is used to connect regions that only have a small gap separating them.

3.2.2 Image segmentation

Image segmentation is an unsupervised classification problem. Each pixel $i \in I$ needs to be assigned to a class. Many different algorithms have yet been proposed to tackle this task, while recently the majority of new algorithms relies on deep convolutional neural networks (CNN). However, in the scope of this project, in which the vision system should not be trained by large annotated datasets, I will give a summary of the state-of-the-art image segmentation and object detection algorithms do not rely on training data sets but are rather cognitive.

Region based methods

K-Means K-Means clustering [52] is an unsupervised classification algorithm relying on the L2-norm to calculate distances. It iteratively assigns each pixel to one of N clusters The number of classes to partition the data needs to be fixed beforehand. The main approach of the k-means clustering algorithm are:

1. Initialization of N cluster centers
2. Assign each data point to the cluster with the closest cluster center
3. Update the cluster centers by calculating the mean of all data points assigned to a cluster
4. Repeat from the second step until the assignments do not change anymore or the cluster update is smaller than a minimum distance ϵ.

For image segmentation, the k-means algorithm can be used to partition the image into N segments. The feature vector for each pixel usually contains the color intensity, but can also contain spatial information. The disadvantage of this method is that the number of segments needs to be defined before the algorithm, but usually is not known a priori, however with the help of clustering indices the optimal number of segments can be determined [62].

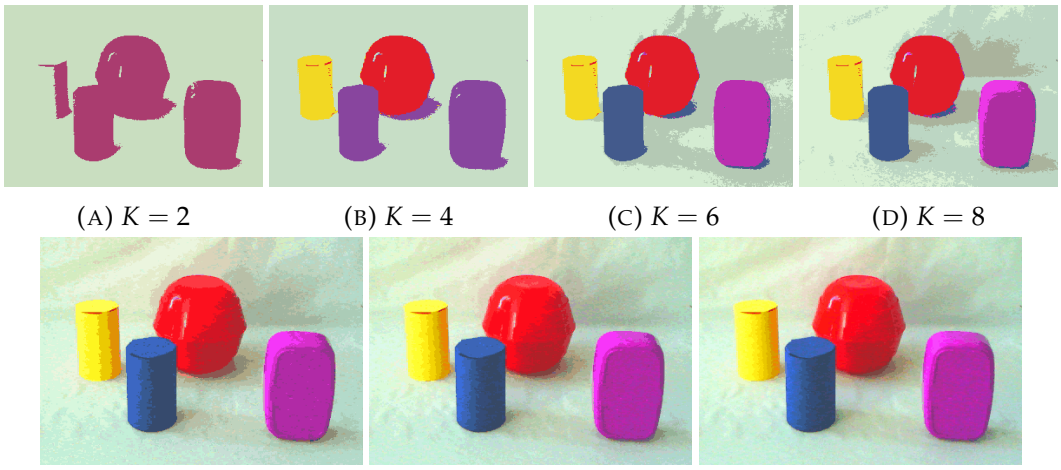
The k-means algorithm is also used for color quantization as a preprocessing step [14].

Meanshift Another unsupervised clustering algorithm that is used in various applications in computer vision [18],[17]. is the mean shift clustering algorithm. The mean shift algorithm is based on the non-parametric density estimator mean shift. The mean shift vector has the direction of the gradient of the density estimate. Therefore it can be used to lead the path to the density maximum. It is computed by

$$M_h(x) = \frac{h^2}{d+2} \frac{\hat{\nabla} f(x)}{\hat{f}(x)}, \tag{3.9}$$

where $\hat{f}(x)$ is the multivariate kernel density estimate at x and $\hat{\nabla} f(x)$ the estimate of the density gradient at x.h is the window size. The density estimate and the density gradient estimates are calculated on a search window $S_h(x)$ which is a hypersphere of radius h centered at x. The mean shift procedure is defined as the following:

1. Compute the mean shift vector $M_h(x)$ on a search window $S_h(x)$



(A) $K = 2$ (B) $K = 4$ (C) $K = 6$ (D) $K = 8$

(E) $K = 32$ (F) $K = 64$ (G) $K = 128$

FIGURE 3.6: Examples for k-means color quantization with different numbers of clusters.

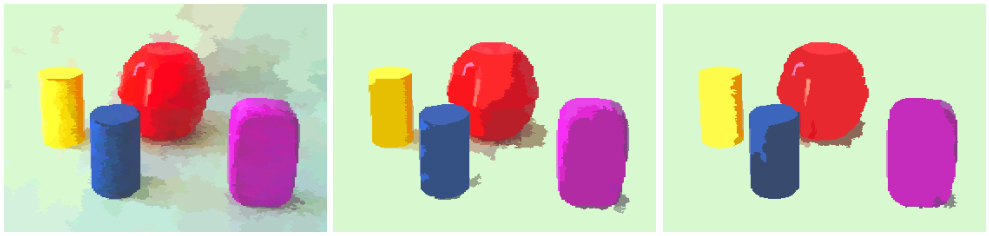
2. Translate the search window $S_h(x)$ by the vector $M_h(x)$

The procedure is applied successive until convergence, which is when the length of the mean shift vector converges to 0 and the search window location is not translated significantly anymore.

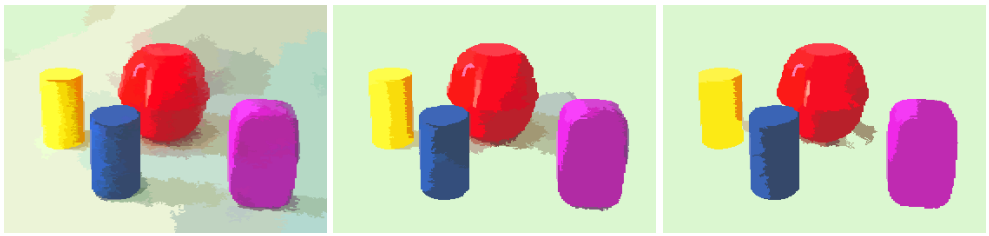
For image segmentation, the mean shift algorithm is both applied on spatial as on range information of the image. In the case of a color image with three color channels, the data points have a feature vector of two spatial coordinate values, the x and the y position of the data point in the image, and the 3 range values, which are the color channels. After a proper normalization with σ_r for the range features and σ_s for the spatial features, this spatial and range features can be concatenated into a spatial-range domain vector. Let $x_i, i = 1...n$ be the data points with the normalized feature vector of the original image, $y_i, i = 1...n$ the points of convergence and $L_i, i = 1...n$ a set of labels. Then the mean shift algorithm can be executed to segment an image with the following steps:

1. For each data point x_i, execute the mean shift procedure until convergence and store the convergence point in y_i
2. Cluster the convergence points y_i by linking together all y_i which have less than 0.5 distance in the normalized spatial-range domain into clusters $C_i, i = 1..m$
3. Assign labels L_i for each data point, by their cluster assignment $L_i = \{m | y_i \in C_m\}$
4. Optionally eliminate spatial regions smaller than M pixels

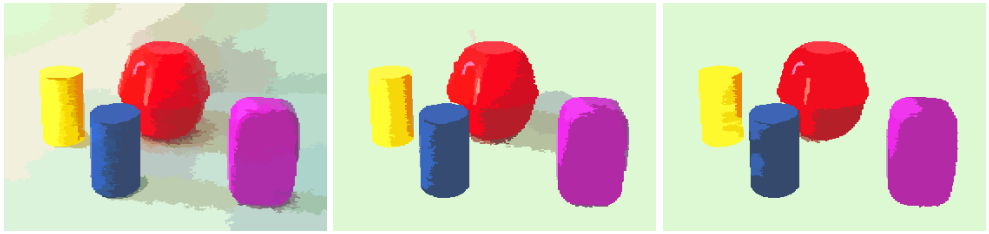
The number of clusters depends on the data normalization factors σ_s and σ_r which need to be adjusted with domain knowledge such as statistical attributes of the image intensities. The advantage of this method in comparison to k-means clustering is that it is a non-parametric, deterministic method that is independent on initialization. The disadvantage is that the arithmetic complexity is higher and therefore the segmentation takes longer.



(A) $\sigma_s = 5, \sigma_r = 5$ (B) $\sigma_s = 5, \sigma_r = 10$ (C) $\sigma_s = 5, \sigma_r = 15$



(D) $\sigma_s = 10, \sigma_r = 5$ (E) $\sigma_s = 10, \sigma_r = 10$ (F) $\sigma_s = 10, \sigma_r = 15$



(G) $\sigma_s = 15, \sigma_r = 5$ (H) $\sigma_s = 15, \sigma_r = 10$ (I) $\sigma_s = 15, \sigma_r = 15$

FIGURE 3.7: Mean shift segmentation with different spatial-range domain normalized data. Note that while the background is detected well in the experiments with $\sigma_r = 15$, the objects are separated into at least two regions.

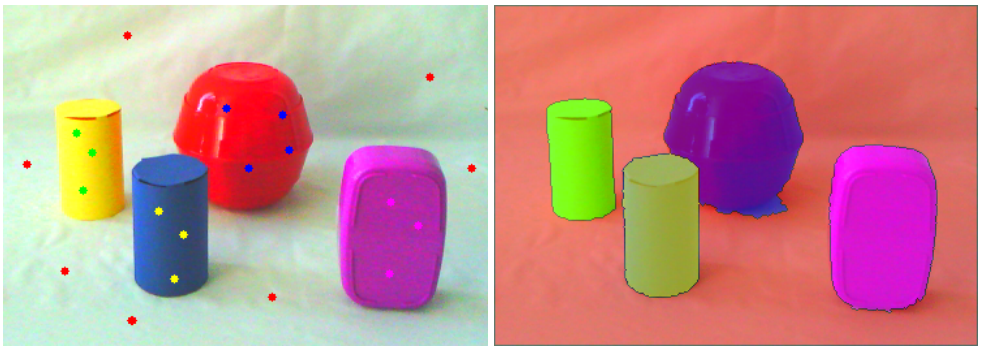


FIGURE 3.8: Example for watershed segmentation of the image. Left: Manually set region markers, right: segmented image. It can be seen that the segmentation is very precise on all objects but the red sphere-like object.

Watershed transformation The watershed transformation [7] is a morphological image segmentation algorithm that relies on the principle of flooding. The image is transformed into a topographic surface and is flooded with water from different locations of the surface with constant vertical speed. At some point of the flooding, two or more floods may merge. The location where they would merge defines the boundaries of our segments. Imaginary speaking we are building a dam at the position where the floods merge. When the entire surface is flooded, the dams describe the outlines of the segmented regions. The method is generally non-parametric, but for object detection it needs two types of markers on the original image: Markers for the outside region of the image, which is the non-relevant background and markers for the inside region of the image, which are the objects that want to be detected.

The watershed transformation is useful when dealing with overlapping objects and is precise on finding the exact borders of these. However, to obtain precise results it needs to be used wit prior segmentation of the image in background and foreground and setting markers for each object. As we will see later, multi-stage approaches can be used to detect the markers automatically.

Graph-based methods

Graph-theoretic methods in image segmentation try to incorporate Gestalt principles [89], explaining the perceptual organization humans perceive, into the segmentation process. Based on early methods [92], later approaches [71][28] showed good results on global image segmentation. The Graph $G(V, E)$ for an image is usually build by the local neighbourhoods of each pixel, so each pixels is a vertex v_i and is has edges to its neighboring pixels. Another approach is to connect each vertex v_i to vertices which are similar to it in feature space. Graph based methods are also executed on images segmented into superpixels.

Normalized Cuts The general principle of normalized cuts is partitioning the image into regions that minimize the *normalized cut* criterion, which maximizes global dissimilarity between the partitions and maximizes local similarity within each partition based on perceptual grouping criteria [71]. The criteria is based on the graph-theoretic problem to divide a graph into two disjoint sets by removing edges. The

total dissimilarity between the sets then can be calculated as the sum of the weights of the removed edges. Since this favors the separation into small, isolated subsets of nodes, a second criteria, the disassociation was introduced, which is computed by the sum of weights that connect the nodes of a partition to the rest of the graph. For two disjoint partitions A, B of a Graph $G(V, E)$ this criteria is defined as:

$$Ncut(A, B) = \frac{cut(A, B}{assoc(A, V)} + \frac{cut(A, B}{assoc(B, V)} \tag{3.10}$$

The problem of finding the optimal partitions is then solved by the following procedure:

1. Setup the Graph $G(V, E)$ for the image where each pixel is a vertex and all vertices are connected initially. The weight for each edge is the calculated similarity between the two pixels.
2. Setup D and W, which are node connection and weight matrices and solve $(D - W)x = \lambda Dx$ for the eigenvectors with the smallest values
3. Separate the graph into 2 partitions with the eigenvector with the second smallest value
4. Recursively repeat steps 1-3 to split the obtained partitions, if necessary.

With the recursive iterations of the algorithms, many objects in an image could be detected. However, in practice, and also in the general idea of the method, it is used better to extract the main object that draws the attention of the image. Furthermore, the solving of the eigenvector equations is arithmetically complex and takes much computing time.

Felzenszwalb-Huttenlocher In contrast to the method based on the normalized-cut criterion, Felzenszwalb et al. [28] proposed a graph-based segmentation method that relies on a predicate D which decides if there should be a boundary between two regions of the image or not. This predicate is measured by the dissimilarity between the elements along the boundary of two neighboring regions. The predicate is true when the minimum difference $Dif(C_1, C_2)$ between two regions is higher that the minimum internal difference $MInt(C_1, C_2)$ of the regions. Using this predicate, the algorithm for segmenting an image in Graph representation $G(V, E)$ with n vertices and m edges and segmentations S is proposed as:

1. Sort edges by increasing edge weight
2. Initialize the segmentation S^0 where each v_i is in its own component.
3. For each $q = 1..m$, construct S^q given S^{q-1}: Given the vertices v_i and v_j that e_q is connecting, If vertices v_i, v_j are in disjoint components in S^{q-1} and $w(e_q)$ is small in comparison to the minimum difference of the two components, merge the components.
4. The final segmentation is $S = S^m$

The algorithm makes greedy decisions, is of less arithmetical complexity than the normalized cuts approach and is able to preserve details in low-variability regions while ignoring detail in high-variability regions.

Supporting concepts

The concept of superpixels was first named in [65]. The idea is to oversegment the image into superpixels which can be used for further segmentation which is less time consuming because of the lower number of elements. Superpixels are small groups of neighboring pixels which should have high intra-region similarity and low inter-region similarity based on the similarities of for example, brightness, color and contour. After first approaches with the normalized-cut algorithm [65], many algorithms to obtain superpixels have been developed [1][46][6]. Once superpixels have been obtained, they can be used just like pixels for image segmentation, but contain more features and reduce the search space for the segmentation. In recent research about image segmentation, methods using superpixels of the images are getting more and more attention.

Region adjacency graphs [87] (RAG) are graphs in which each vertex is a region and it is connect by edges to their adjacent regions. They are used as a common data structure for image segmentation, normally in a subsequent step after a oversegmenting segmentation, as for example with superpixels. The principal advantage of these structures is that they give more weight to the adjacency relationships between regions than the usual methods. [85]

Multi-stage approaches

As we have seen in the examples for the already discussed image segmentation approaches, these can already give fairly good results. However, many for them have a specific goal and are often not useful for a more global, automatic image segmentation task. Therefore continuous research on multi-stage image segmentation approaches which intent to use the advantages of different methods is done. In the following I will give examples of current state-of-the-art multi-stage approaches.

In [4] a contour detection algorithm gPb and a image segmentation method using the gPb contour detector, an Oriented Watershed Transform and an Ultrametric Contour Map. The method provides state-of-the-art results on three image segmentation datasets. In [82], the mean shift segmentation method is used as a preliminary step to create an oversegmented segmentation. The RAG of this segmentation is then used with the normalized cuts algorithm in order to find high-level partitions.

3.3 Object description

When the image is segmented and a collection of objects with their boundaries have been identified, another step to understand the scene is to describe each object. There are many ways to describe a object on different levels of detail. The way how humans describe objects is the most high level, using either uniquely identifying terms like "Red cube" or terms that describe parts or attributes of the object like "Metallic red thing". At this stage, the grounding between "red" and the numerical color intensity values has already happened. As described earlier, for this to happen, we need a more abstract layer of description to enable the grounding. In this section I will explain how we can use continuous low level features to describe a object that can be derived as numerical values from the image intensities.

3.3.1 Object features

An object can be described by many different attributes, but for the context of this thesis I will give an overview over existing visual descriptors for size, color, shape and texture.

Color

To numerically describe a color, a set of different color models have been introduced. The most common ones represent colors by a tuple of triples or quadruples. Each color model has a different color spaces which define the implementation of the color model. These color spaces vary for example in the way how color distances are calculated. As a knowledge basis for color descriptors I will summarize the most commonly used color models in computer vision.

RGB The RGB (**R**ed-**G**reen-**B**lue) color model uses additive color mixing, The general concept is that by summing a red, green and a blue light source with different intensities every color can be created. The most common color space for this model is the sRGB (standard Red-Green-Blue) color space which was developed by Microsoft and HP.

YUV The YUV color model represents colors by a luma component Y and two chroma components U and V. Luma stands for the brightness of a color, the U and V components respectively represent blue- and red-luminance of the color. YCbCr is color space of this model where C_b is representing the blue-yellow part of the color and C_r the red-cyan part.

CIELAB Similar to the YUV model, the CIELAB, often just abbreviated as Lab color space, is composed of three components, where the L component represents the lightness from black (0) to white (100), the a component the luminance of the color between green and red, having its neutral gray value at 0 and the b component the luminance of the color between blue and yellow, again having at 0 a neutral grey value.

HSV The HSV (**H**ue-**S**aturation-**V**alue) color space was developed to be closely aligned with the human perception of color-making attributes. The hue part of this color space is a circular scale, while saturation and either value or lightness are continuous values in the range of 0 and 1. In the hue dimension, 0 ° represents the primary red color, 120° the primary green color, 240° the primary blue color and approaching to 360° the primary red color again, as it is a radial scale.

Color descriptors A color descriptor is used to numerically describe the color structure of an image or an image region. A good color descriptor should be illumination invariant and have high discriminative power. In the following I will discuss different approaches.

The most simple way to describe an region is by its *mean color value*. To achieve illumination invariance, color spaces that use a single component for brightness are favorable, as the YUV, CIEALB or HSV color spaces.

Another way to represent the color of a region is by computing a color histogram. Color histograms M_c represent frequencies of image intensities on the color channel c computed over the pixels belonging to the object. The range of intensities is divided into m bins $k \in \{1..m\}$. The number of pixels that have intensities falling into each bin $M_c(k)$ is counted using a function $h(i_c(p))$ that assigns the intensity i_c of a pixel p to a bin k.

A more mathematical approach to color descriptions is introduced by *generalized color moments* [55]. By regarding RGB triplets as data points coming from a distribution, it is possible to define generalized color moments:

$$M_{pq}^{abc} = \int\int x^p y^q [I_R(x,y)]^a [I_G(x,y)]^b [I_B(x,y)]^c dxdy \tag{3.11}$$

$M_p^{abc}q$ is referred to as a generalized color moment of order $p+q$ and degree $a+b+c$. Combining different moments, one can normalize them so that they are invariant to light intensity changes and shifts as well as to color intensity changes and shifts. These combinations are called *color moment invariants*

The SIFT descriptors [50] are are based on the *scale invariant feature transform* and are a key point descriptors used for object recognition based on key features. Their color description capabilities are restricted, since they are not invariant to light changes. The SIFT color descriptor used in combination with other color descriptors has created a family of SIFT-based color descriptors. In an experimental evaluation [69], *OpponentSift* achieved best results on color descriptor based object recognition. It describes all color channels in the opponent color space:

$$\begin{pmatrix} O_1 \\ O_2 \\ O_3 \end{pmatrix} = \begin{pmatrix} (& \frac{R-G}{\sqrt{2}} \\ & \frac{R+G-2B}{\sqrt{6}} \\ & \frac{R+G+B}{\sqrt{3}} \end{pmatrix} \tag{3.12}$$

Size

Intuitively the description of the size should be straight forward. However, without stereo vision, estimation of distance is a non-trivial problem. I will first discuss common ways to describe the size of an object and then comment on how to estimate its real size by estimating the distance.

The easiest way to describe a image region by size is by height and width. One can either calculating the bounding box of the image region, which is the smallest rectangular with sides parallel to the image boundaries that includes the entire object. However, this approach does not take into account the rotation of the object. Therefore one can also compute the rotated bounding box, which is the smallest rectangular that contains the region without the constraint of parallel lines to the image boundaries. Given a bounding box around the image, its height, width and area can be calculated. A more precise way to calculate the area of a region is to calculate its contour area. This can be done either by counting the pixels of the region or by calculating the contour using Green's theorem: Let D be the region and C the

list of the closed contour with counterclockwise points of the contour. Then

$$A_D = \int_C F\,ds = \frac{1}{2}\int_C x\,dy - y\,dx \tag{3.13}$$

where $F(x,y) = \left(\frac{-y}{2}, \frac{x}{2}\right)$, can be used to calculate the contour area.

Real size estimation The calculation of the distance of an object with a single image is a challenging task. If we do not have information about the camera location and angle, precise mathematical calculations are not possible. If we know about the camera location in 3D-coordinates and the Y-rotation of the camera, one con estimate an objects height by a sequence of calculations based on the angle invariance, as described in [29].

Shape

As the shape of an object we call the outer form of the boundaries of the object. The boundaries are described by the outer edges of the object. In geometrical terms, edges are a set of attached lines. In general, shapes can be described by a number of different atomic features like the number of corners, the number of straight lines, the number of curved lines or the number of parallel sides. Primitive shapes can be classified into polygons and ellipses. Polygons are composed only by straight lines and can be further classified based on their number of corners such as triangles, quadrilaterals or pentagons.

Shape Descriptors Although the notion of shape for a object might seem intuitively well defined, it may have different meanings in different contexts. Specifically, shape descriptors can be classified in contour-based and region-based descriptors. Shape descriptors should be invariant to scale, translation and rotation.

A simple way to describe basic shapes is by counting the corners of the object contour. This can be done by approximating the objects contour and counting the necessary points to describe the object. Directly related to the corners of the contour of an object, the inner angles calculated by the two neighboring vectors of a corner are a further simple shape descriptor.

Region-based shape descriptors are usually based on image moments: A general shape descriptor is provided by the Hu moments [38]. These are a set of image moments especially designed for shape and region description. Image moments are a weighted average of image pixel intensities. The Hu moments are based on the normalized centralized image moments which are computed as following:

$$\eta_{ij} = \frac{\mu_{i,j}}{\mu_{00}^{(i+j)/2+1}} \tag{3.14}$$

where $\mu_{i,j}$ are centralized moments defined as

$$\mu_{ij} = \sum_x \sum_y (x - \bar{x})^i (y - \bar{y})^j I(x,y) \tag{3.15}$$

The normalized centralized image moments are invariant to scale and translation, but not to rotation. Therefore the 7 Hu moments were introduced. They can be calculated as following:

$$
\begin{aligned}
h_0 &= \eta_{20} + \eta_{02} \\
h_1 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
h_2 &= (\eta_{30} - 3\eta_{12})^2 + (3\eta_{21} - \eta_{03})^2 \\
h_3 &= (\eta_{30} + \eta_{12})^2 + (\eta_{21} + \eta_{03})^2 \\
h_4 &= (\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\
&\quad + (3\eta_{21} - \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
h_5 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2 + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03})] \\
h_6 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] \\
&\quad + (\eta_{30} - 3\eta_{12})(\eta_{21} + \eta_{03})[3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2]
\end{aligned}
\tag{3.16}
$$

A different approach is based on the Zernike polynomials. The Zernike moments [19] can be derived by integral calculation from the Zernike polynomials, but can also be calculated by normalized centralized moments. As the explanation of the direct calculation of the Zernike moments requires the introduction of the concept of Zernike polynomials, the formulas for the Zernike moments are ommitted.

Lastly, another approach to shape description is to describe the shape by its contour. A simple contour based method is polygon approximation, in which the shape is approximated by a polygon. There are merging and splitting approaches to polygon approximation. One of the state-of-the-art polygon approximation algorithms is the widely used Ramer–Douglas–Peucker algorithm [22] which recursively divides the line and removes a point if it has less than a certain distance ϵ from a line that connects the neighbouring points.

3D shapes The concepts of shape is also extended to three dimensions. A three-dimensional (3D) shape can be described by a composition of two dimensional shapes. If a two-dimensional (2D) part of the composition is flat, it is called a face of the object. Compositions of primitive 2D faces can be classified into broad categories of 3D depending on the parts they are composed of: Objects consisting of flat faces and straight edges and sharp vertices are called polyhedra. Special cases of polyhedra a for example a cuboid, which consists of 6 quadrilateral faces or a cube which is composed by 6 squares as faces, having right angles on vertices.

In the given application of VQA with 2D images, a challenging task is to derive the 3D shape of an object from its 2D shape. Furthermore the outer 2D shapes of 3D objects are often not sufficient to discriminate the 3D shape.

Texture

Another way to describe and identify objects is by their texture. Generally speaking the texture is described as the visual or tactile surface characteristics and appearance of an object. Texture can be seen as two orthogonal properties, spatial structure, which describes the pattern and contrast, which is the amount of local image texture.

In this section I will explain the most common descriptors to discriminate different textures.

Color distribution A simple approach to analyze an objects texture is to calculate the distribution of colors of the object. This is frequently done channel-wise by calculating the histogram of the intensities per channel. The HSV color space has proven to be useful for this as each channel describes independently a different semantic attribute. Therefore the histogram of the hue channel gives insight about the distribution of different color hues in the object and can be used to describe if the object is rather single colored or build of multiple colors, the histogram of the saturation channel expresses the intensity of the colors in the object and the histogram of the value channel gives insight over the brightness of the color. A distribution with both a high amount of high as of low brightness colors can for example be an indicator that the object is strongly reflecting, since other objects that are reflected in the object usually have very low brightness on the objects surface and regions with very high brightness are reflected light sources. However, since the histogram does not give insight about the spatial distribution of the colors, this descriptor has limited discrimination capacity for textures.

Gabor Filters Gabor filters [30] are filters used to describe textures. The filters are build by a Gaussian kernel modulated by a sinusoidal plane wave. A typical Gabor filter bank is a set of Gabor filters consists of rotated filters to detect edges, gradients or circles. A problem of Gabor filters is that there exists no globally applicable Gabor Filter Bank and the Gabor Filter Banks are problem-dependent.

Haralick features The Haralick features These contain information about textural characteristics like homogeneity, gray-tone linear dependencies, contrast, number and nature of boundaries present, and the complexity of the image. They are calculated by calculating the Gray-Level Co-occurrence Matrix (GLCM). The 14 Haralick features are the Angular Second Moment, Contrast, Correlation, Sum of Squares: Variance, Inverse Difference Moment, Sum Average, Sum Variance, Sum Entropy, Entropy, Difference Variance, Difference Entropy, Info. Measure of Correlation 1, Info. Measure of Correlation 2 and Maximum Correlation Coefficient. [36].

Local Binary Patterns Local Binary Patterns [59] are visual descriptors on gray scale images that are invariant to rotation and monotonic transformations. One Local Binary Pattern value is calculated on P neighboring values in a radius R of the center pixel g_c. On a local image the rotation-variant descriptor can be computed by

$$LBP_{P,R} = \sum_{p=0}^{P-1} s(g_p - g_0)2^P \tag{3.17}$$

where s(x) is the sign function. A common way to use them to describe the overall texture of an image or an object is to calculate a LBP histogram.

3.4 Main challenges

After having discussed the theoretic concepts in image understanding in computer vision, I will discuss the main challenges I faced during the implementation of the vision system.

Occlusion Humans intuitively are able to detect if an object is occluded by another one and if, estimate its shape. However, this requires a reasonable amount of domain knowledge and reasoning capabilities, that a computer vision system cannot represent initially. Especially detecting occlusion of objects with similar visual attributes is a task that is challenging for any image segmentation algorithm.

Level of abstraction Another challenge including semantics and domain knowledge is the detection of instances of objects. As there is concept of an object is not well-defined, domain knowledge and semantic, situation dependent reason is required to detect the right level of abstraction when labeling detected regions as objects. A car for example can be both identified as one object as also as a composition of many object like its wheels, doors, windows and so on. The choice of the right level of abstraction clearly depends on the application. Even if in the performed experiments in this thesis the definition of an object is well defined by the sole existence of geometric solids, the implementation of this knowledge into the computer vision system still is a challenging task.

2D to 3D shape Lastly, the derivation of the 3D shape from the 2D contour is a complex task. Since even if the full object is seen in the image, parts of it are always self-occluded. Additionally, small errors in the region segmentation of an object due to noise increase the difficulty of accurate 3D model estimation. A robust description of the 2D shape is therefore a key requirement for shape estimation of geometric solids.

Chapter 4

Contributions

This chapter gives a summary over the contributions of this thesis.

4.1 An algorithm for object detection and description

The main contribution of this work is the proposal of an algorithm for object detection and description with minimal bias and no training. The algorithm consists of foreground detection, color image segmentation with the mean shift segmentation algorithm [17] and instance segmentation with a watershed transform [7]. The detected objects are described with a number of numerical features that can be linked to semantic symbolic classes describing size, shape, color and material of the objects.

4.2 An end-to-end VQA system for robots

The second contribution is the embedding of the proposed algorithm in a end-to-end VQA system that can be deployed to embodied agents. The natural language processing part of the system was mostly developed by the artificial intelligence lab of the Vrije Universiteit Brussel (VUB). The system will be able to answer a broad range of compositional basic questions on scenes with basic geometric solids in different colors, shapes, sizes and materials. The VQA tasks include various aspects of visual reasoning including attribute identification, counting, comparison, spatial relationships and logical operations.

4.3 A CLEVR-like dataset with real images

The last contribution is a dataset to test the developed VQA system and especially the image segmentation algorithm on images taken by the NAO robot. Therefore I provide an annotated dataset with 150 images with similar scenes to the CLEVR dataset [41], offering the challenges of real life environments as varying light conditions, image quantization noise and objects with small artifacts.

Chapter 5

Implementation

This chapter focuses on the implementation details of the proposed system. The developed system will have its core part in the image segmentation and object description, while the language parts will be described in less detail because they were not developed by the author of this thesis and just form part of the overall necessary framework. Furthermore, the developed vision system is meant to be used for different use cases.

In the first section I will shortly describe the NAO robot, whose vision system will be used in the VQA system.

The second part of the chapter gives an overview over the implemented VQA system and how the components are connect. The third section gives a detailed summary of the implemented vision system, the core part of the implementation of this thesis. Finally, in the fifth section I will explain the details of how the outputs of the vision system can be used link low-level object descriptors to semantic concepts.

5.1 The NAO robot

The NAO robot was developed by Aldebaran Robotics. It is a humanoid robot that has been designed for educational and research purposes. It is 58cm heigh and weights 5.2kg. It has 25 degrees of freedom, 7 touch sensors, 4 directional microphones and 2 2D cameras. One camera is located at the top of the head, looking forward, the other camera is located at the bottom of the head, facing downwards to the legs. Their view angles are not overlapping.

In this thesis, we will only use the top camera of the robot with a resolution of 640x480 pixels. The camera unfortunately is very sensitive to lighting changes and reveals bad capabilities of representing captured images under bad light conditions.

The robot can be seen in figure 6.4.

5.2 VQA system

The VQA system takes a question and a image as the input and returns the answer to the question, if the scene could be successfully analyzed and the question successfully parsed and mapped to the scene. The main steps are visualized in figure 5.1.

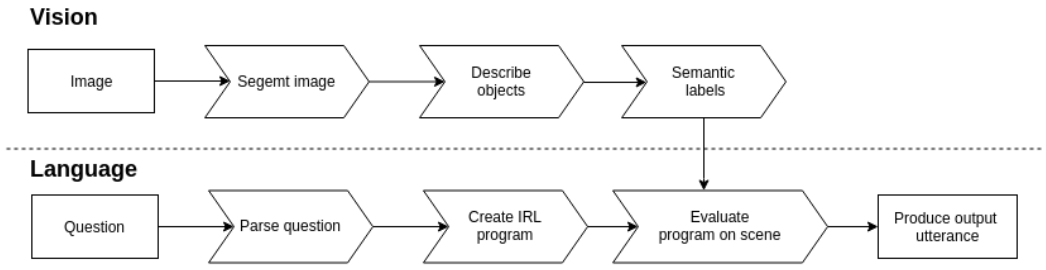


FIGURE 5.1: Overview over the implemented system

5.2.1 Overview

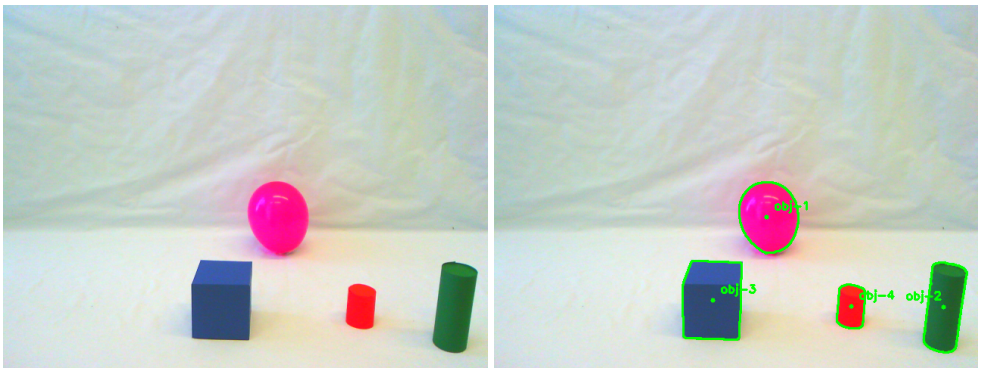
In the following paragraphs I will describe an example to demonstrate the processes happening in the implementation. For this, I will use an example setup which I will explain briefly: The image describing the scene can be seen in figure 5.2a. The question to be answered is **"Are there any other large things that are the same material as the cube?"**.

Firstly, the vision system analyzes the captured image. The output of the analysis is a list containing the detected objects with their position and their object descriptors. Furthermore, a annotated image is created for visual guidance. (see figure 5.2b). Tables 5.1 and 5.2 show the main object descriptors returned for each object. Complex descriptors consisting of more than one value are not displayed.

Then the question is parsed with a FCG grammar, analyzing the syntactic structure of the question. Since the grammar has its constructions enhanced with the semantic meanings, the corresponding functional IRL program can be derived directly from the syntactic structure. The syntactic structure is visualized in figure 5.3. The IRL program that needs to be evaluated to answer the questions is displayed in figure 5.4.

Now the last step before the IRL program can be evaluated on the list of detected and described objects to obtain an answer to the question is to link the semantics of the IRL program to the detected objects. In other words, the detected objects need to be enhanced with semantic labels which describe their attributes and spatial relationships. This task is a typical classification problem, where the classifier uses the object descriptors as the input vector and the output is the semantic class that the object belongs to. In a later section the basic classifiers used for the VQA experiments of this project will be briefly discussed. Additionally I will show how these links can be created by emergent incremental learning by agents playing language games. For this example, the objects with their semantic attributes are shown in table 5.3.

Then the IRL program can be executed: First `get-context` gets the semantically described objects as the context. Then `filter` filters the objects for objects that have shape *cube* and tests if only one object of the context matches the filter with `unique`. Then the `same` program is initialized with its parameters: The object returned by the `unique` program - the reference cube -, the attribute *material* that will be tested to be equal. This returns the set of objects that have the same material as the reference cube, *paper*: *obj-2* and *obj-4*. This set will be filtered by `filter` for shape *thing* which is the generic category for all objects and size *large*. The resulting filtered set only contains *obj-2*. Then, to finally answer the question, `exists` tests if the filtered set



(A) Raw image (B) Annotated image

FIGURE 5.2: Image for the described example. There are four objects: A large blue paper cube, a large magenta plastic sphere, a small red paper cylinder and a large green paper cylinder. (from left to right)

id	x	y	w	h	corners	circle_d	area	bb-area	size-ratio
obj-1	360	280	78	94	14	0.855	5719	7332	0.83
obj-2	594	399	118	47	8	0.94	5048	5542	0.40
obj-3	289	390	103	78	5	0.99	7607	8049	0.75
obj-4	472	398	58	36	8	0.94	1860	2074	0.62

TABLE 5.1: First part of the object descriptors for the example scene: id of the object, x and y are the pixel coordinates in the image, w describes the width of the object, h the height of the object. Corners describes the number of corners after simplifying the contour, circle_d the hamming distance to its enclosing circle, area the pixel-area of the contour, bb-area the area of the bounding box and size-ratio the ratio of width and height.

contains at least one element. In the given scene, this is the case and therefore the returned answer is **"Yes"**.

id	hue	saturation	value	p_whites	p_blacks
obj-1	163	235	254	0.2%	0.0%
obj-2	68	163	106	0.0%	0.0%
obj-3	110	149	131	0.0%	0.0%
obj-4	179	232	255	0.0%	0.0%

TABLE 5.2: Second part of the object descriptors for the example scene: id of the object, mean hue value of the object (on a radial scale from 0 to 180), mean saturation value (linear scale from 0 to 255), mean value (linear scale from 0 to 255), p_whites is the percentage of white pixels in the object, p_blacks is the percentage of black pixels in the object.

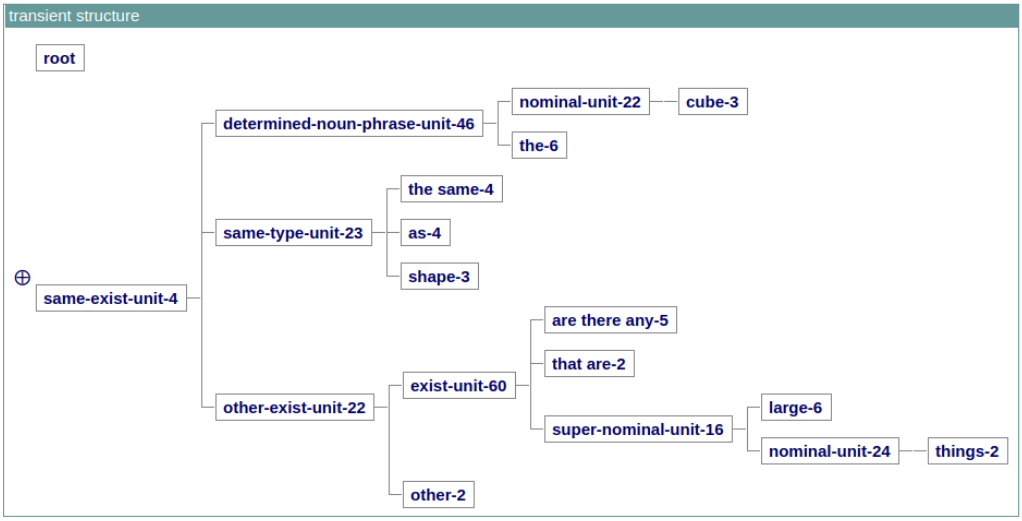


FIGURE 5.3: Resulting transient structure for the example sentence

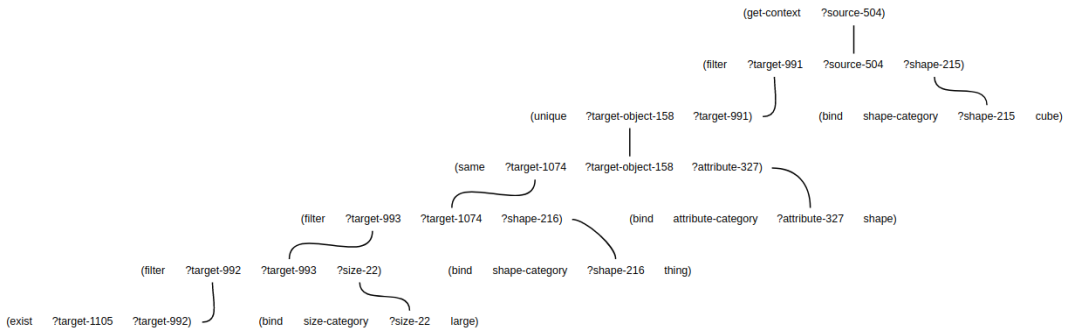


FIGURE 5.4: Resulting IRL program for the example sentence

id	size	shape	color	material
obj-1	large	sphere	magenta	plastic
obj-2	large	cylinder	green	paper
obj-3	large	cube	blue	paper
obj-4	small	cylinder	red	paper

TABLE 5.3: Semantic labels for the attributes size, shape, color and material for the detected objects

5.3 Vision system

The vision system developed in this project has two main responsibilities: object detection and object description. Since the captured pictures of the robot usually are of a higher level of detail than necessary for the task, preprocessing the image is an import step to prepare it for the object detection task. I will first discuss the three steps of preprocessing, then go through the processes of the object detection and lastly explain how the object descriptors are extracted.

I should be remarked here that all the methods used in the system are deterministic, therefore the outcome of the segmentation run on the same image of two sequential execution is identical. Furthermore the algorithm is not parameter free. Some of the used methods require parameters to be set. In the following I will only comment and describe the parameters that significantly change the final result, e.g. the detection of false positives or the accuracy of the boundaries. The other parameters are mostly the recommended parameters from the authors of the algorithm.

5.3.1 Preprocessing

The preprocessing of the captured image has three goals: Remove noise, enhance edges and reduce color space. This is necessary to obtain a clean view of the scene, highlighting the main aspects of the image. I will explain shortly which method was used for each goal and why.

Noise removal and edge enhancement

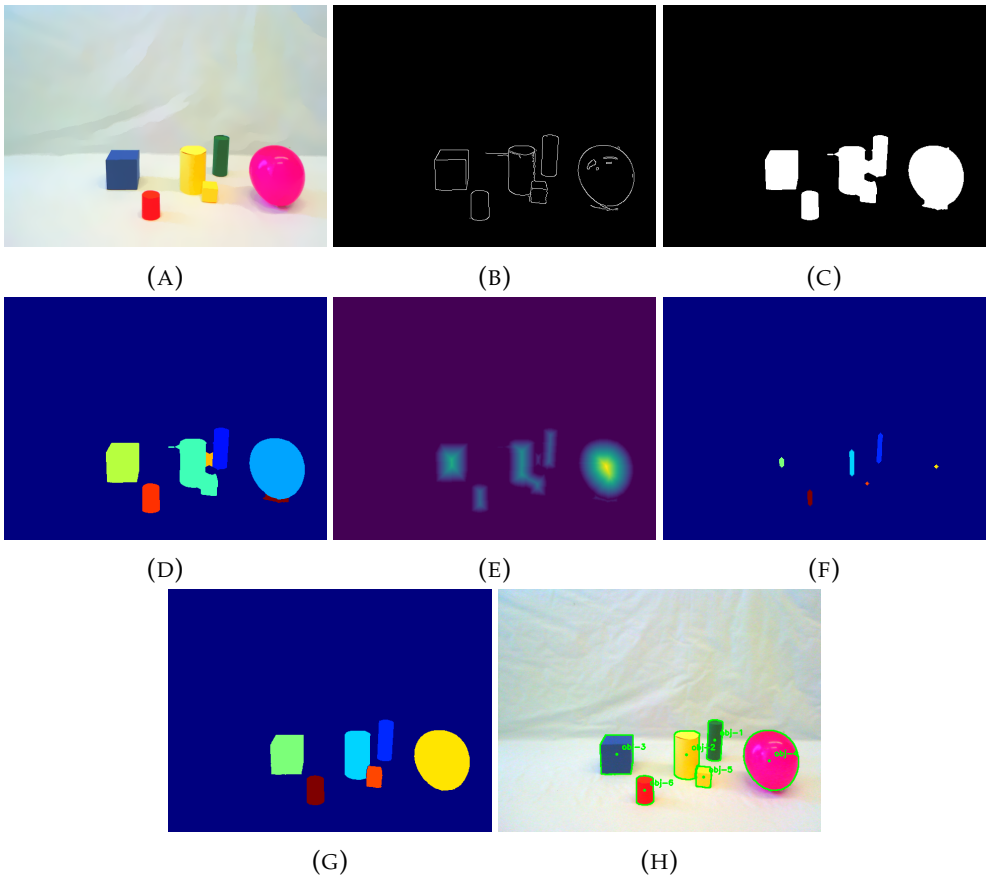
To reduce noise introduced by bad light conditions or quantization errors of the camera, I am using Non-Local Means Denoising (NL-Means Denoising) [11]. Then, to reduce the color space and to enhance object edges, pyramid mean shift filtering [18] is applied.

5.3.2 Object detection

The object detection on the denoised and prepared image is they key part of the system. It first detects the sure foreground areas by detecting edges, finding contours and morphological operations. Then the extracted foreground will be presegmented by colors. With the use of a distance transform and local maxima detection the markers for the watershed algorithm are set. The watershed algorithm then segments the regions into the final objects, separating even overlapping same color objects. In the following I will discuss each stage of the method and explaining the reasoning behind the implementation.

Foreground detection

After the image was prepared by the described preprocessing steps, the foreground regions need to be detected. The main assumption on the environments that the system should be working with will be that the image background is of rather uniform texture with no clear edges. With the detection of the edges with the Canny edge



(A) (B) (C)

(D) (E) (F)

(G) (H)

FIGURE 5.5: Intermediate steps of the image segmentation. 5.5a) First the image is preprocessed by removing noise and enhancing edges. 5.5b) Then the edges are extracted. 5.5c) After applying morphological operations to close contours, the foreground is extracted. 5.5d) The mean shift segmentation algorithm is used to pre-segment the foreground by colors. 5.5e) Distance transform is applied to each connected color segment. 5.5f) Local maxima in the distance transform representation are used as the markers for the watershed image segmentation. 5.5g) Segmented image after watershed transform. 5.5h) The final detected and annotated objects.

detector [12], the strongest edges will be found which represent the contours of the foreground regions. In order to find all the foreground regions by a contour finding algorithm [81], it is first necessary to close the contours. This is done by applying a morphological closing operation to the detected edges. After the contours have been detected, small regions below a threshold $t_{foreground}$ will be discarded, the remaining contours will be flood-filled to obtain the foreground matte.

Mean shift segmentation

The detected foreground is then segmented on the spatial-range domain into broad components. Although the segmented regions from this step already return good results, in order to separate overlapping objects of similar colors, the color segmented foreground will be further segmented by a watershed transform [7].

Finding the markers

In order to use the watershed transform segmentation in an automatic way it is necessary to initialize markers on the image to roughly define the background parts of the image, the unknown parts of the image and markers for each object that should be detected. This is achieved by applying a distance transform on each segment of the segmentation obtained by the mean shift segmentation. The markers are then found by calculating the local maxima of the distance transform with the L2-norm and a 5x5 mask [9] with a minimum distance threshold $t_{minDistance}$ and clustered by similarity in spatial-range domain into clusters C. After experiments with different thresholds, the value 5 was found to give good results. Note that the because of the minimum distance threshold, in small and weakly connected regions no markers will be detected. Since those regions are still part of the detected foreground, this does not mean, that those regions which do not contain markers, will not be part of the detected foreground objects.

Region detection

Now, with the set of background markers B, the set of unknown markers U and the set of labeled foreground markers F where $L_i = \{j | F_i \in C_j\}$, the watershed transform [7] is applied to find the object regions. Then for each segmented region their contours are found with a more precise algorithm [83] to detect an appropriate number of points shaping the object.

5.3.3 Object description

After the image segmentation returned the detected objects, each object is described by numerical features including its color, shape, size, texture and the objects location.

Color

In order to describe the color of the object I am calculating the mean color value of the object in the HSV color space, since its hue channel is very discriminate in a

similar way as humans perceive colors. The hue channel of the HSV color space is on a radial scale, therefore we need to calculate the mean value of circular quantities $\alpha_1 ... \alpha_n$ as following:

$$\overline{\alpha} = \text{atan2}\left(\frac{1}{n}\sum_{j=1}^{n}\sin\alpha_j, \frac{1}{n}\sum_{j=1}^{n}\cos\alpha_j\right). \tag{5.1}$$

The saturation and value channel values can be calculated normally. Additionally to the mean color, I am also calculating the normalized color histograms. Therefore, the resulting color descriptors are

1. Mean color in HSV space $\overline{c_i}$
2. Color histograms M_c for $c \in [1, 2, 3]$ for respectively H,S,V color channels

Size

As size descriptors, the contour area, the rotated bounding rectangle area and the width and height of the rotated bounding rectangle are extracted. The contour area is calculated using Green's theorem [66]. The resulting size descriptors are

1. Contour area A
2. Rotated bounding rectangle area A_{bRect}
3. Width and height of bounding rectangle w, h

Shape

In order to describe the shape of the object, it is necessary to simplify the contour by approximating it. This is done by the Douglas-Peucker algorithm [22]. The algorithm needs to be passed one parameter, ϵ. The usual way to calculate ϵ is by computing the contour perimeter or arc length and then multiplying it by a factor which controls the precision of the contour. As we want to maintain corners as much as possible, a very low factor 0.01 was chosen. The algorithm returns the list of points of the contour. The length of the list is the approximate number of corners of the shape. Furthermore the inner angles $alpha_i, i = 1...n$ where n is the number of corners and $\vec{a_i}, \vec{b_i}$ the vectors to the neighboring corners are calculated as following:

$$\alpha_i = arccos\left(\frac{\vec{a_i} \cdot \vec{b_i}}{|\vec{a_i}||\vec{b_i}|}\right) \tag{5.2}$$

. Lastly the 7 commonly used Hu moments [38] (see paragraph 3.3.1) are extracted on the object mask. The resulting shape descriptors are

1. Number of corners n
2. Inner shape angles $\alpha_i, i = 1...n$
3. Hu Moments $\phi_i, i = 1...7$
4. Contour area - bounding rectangular ratio r_{bRect}

Texture

The texture descriptors computed in the object description system are a histogram of Linear Binary Patterns and the Haralick features [36]. The Linear Binary Patterns are computed with the 8 points P in a radius $R = 1$ for each pixel in the object and the summarized in a normalized histogram $LBP(k)$ where k the number of bins $k = 1...P + 1$. The Haralick feature are computed on the bounding box of the object with an additional flag to ignore 0 values to treat them as the background. The resulting texture descriptors are

1. Histogram of Linear Binary Patterns $LBP(k), k = 1...9$
2. Haralick features $f_i, i = 1...14$

Location

The location of the object will be described as the coordinates c_x, c_y of the center pixel of the object. The center pixel of the object is computed by finding the minimum enclosing circle around the object.

5.4 Grounding semantic concepts

In order to use the numerical object descriptors to assign semantic classes to the objects, we need to define how the grounding can be done. Clearly, to find the most discriminant object descriptors and a corresponding classification system for each of the attributes, one could employ a supervised learning approach using an appropriate dataset for each of the classifiers. However, as the system developed in this thesis is meant to ground semantics in later stages of research by emergent incremental learning through communication, this approach would not fit the methodology of this thesis. Therefore I established manual discrimination rules based on the domain knowledge from example images. In further stages of the project, these could be used as a initial prototype for the incremental grounding of the semantics.

The semantic grounding on the perceived color values of the detected objects is done based on the mean HSV color value. The grounding is done based on color prototypes with ranges, as I found this to be an adequate model for further evolution. The implementation for this is done with fuzzy rules: By defining a set of prototypes with the prototype value $c_p = (h, s, v)$ and a lower $c_p^{lower} = (h^{lower}, s^{lower}, v^{lower})$ and an upper bound $c_p^{upper} = (h^{upper}, s^{upper}, v^{upper})$, for a given color the membership to each of the semantic classes can be computed.

The task of classifying shapes, as explained in section 3.3 can be approached in many ways. After performing several tests on the discriminating capabilities of the different descriptors, I came to the conclusion that a simple, intuitive way provides better results and better explainability. As the outer 2D shape varies significantly for different perspectives of a 3D object, semantic rules closer to the way humans understand and ground 3D shapes need to be applied. In the implemented system I am using a rule-based approach to model how one could reason the link between 2D shape of 3D geometric solids. The definition of a cube for example is that it is a region of space formed by six identical square faces joined along their edges. Three edges join

at each corner to form a vertex. Therefore one can conclude that a cube has 8 corners, although depending on the perspective, a maximum of 7 corners can be seen. Therefore, when only having to discriminate between basic geometric solids like cube, sphere and cylinder, as in our experiments, a prototype based discrimination based on the number of corners can be done. With the explained contour approximation, cylinders and spheres have significantly more corners, as round segments are represented by many corners. Additionally, the shape is not easy to estimate if parts of the object are occluded. To estimate if an object is partly occluded, I am calculating the ratio of the contour area of the object $A_{object}^{contour}$ to the bounding box area A_{object}^{bb} of the object:

$$\varrho_{area} = \frac{A_{object}^{contour}}{A_{object}^{bb}}. \tag{5.3}$$

Then I define that an object is occluded if $\varrho_{area} < 0.5$. As one can see less corners in a occluded object the rule-based formula for shape estimation is used as following: Given the number of corners $n_{corners}$:

$$L_{shape} = \begin{cases} cube, & \text{if } n_{corners} < 8\varrho_{area} \\ cylinder, & \text{if } n_{corners} > 8\varrho_{area} \ \ \& \ \ d_{circle}^{hamming} > 0.85 \\ sphere, & \text{otherwise} \end{cases} \tag{5.4}$$

where $d_{circle}^{hamming}$ is the hamming distance of the shape to its enclosing circle.

Intuitively, the semantic labeling of the size of objects should be a straight-forward task. In practice, many aspects have to be taken in consideration. The semantic labels *small* and *large* are usually not absolutely, but relatively grounded. This means, that an object can be small in one context, but large in another. Therefore semantic size labeling in practice needs domain knowledge. Another aspect to consider is the occlusion of objects. If only a part of a object is visible, domain knowledge is necessary to estimate the size of the full object. As the implementation of this is highly complex, in this implementation, the bounding box of the object is used for describing the size of the object. With this, occluded parts of the object are estimated by symmetry. Then the distance to the object is also an important factor. Objects further away seem smaller and their bounding boxes have a smaller area than objects of the same real size closer to the camera. Without stereo vision, distance calculations from 2D images can be done only on a very approximate level. As it is sufficient for our experiments, objects higher in the image are considered further away, while objects lower in the image are considered closer to the agent. Therefore, a very simplified depth estimation is used:

$$d_{object} = \frac{I_h - c_y}{I_h}, \tag{5.5}$$

where I_h is the height of the object and c_y the y-coordinate of the center point of the object. Note that this is a relative depth between 0 and 1. Then a simple area estimation can be computed by

$$A_{object}^{estimated} = d_{object} A_{object}^{bb} \tag{5.6}$$

where A_{object}^{bb} is the area of the bounding box of the detected object. For the classification of an object, two different approaches have been tested: An absolute threshold

and a clustering approach. In the absolute threshold approach, the semantic label is obtained by

$$L_{object}^{shape} = \begin{cases} small, & \text{if } A_{object}^{estimated} < t_{large} \\ large, & \text{otherwise} \end{cases} \tag{5.7}$$

where t_{large} is the threshold to be defined. The clustering approach is more flexible and also considers the aspect that frequently *small* and *large* are used in comparison. The principle is simple: Fit the k-means clustering algorithm with two clusters to the estimated object sizes and label the objects assigned to the cluster with the smaller cluster center value with *small* and the objects assigned to the other cluster with *large*. However, this method does not work out of the box for scenes with only one object or scenes in hwich all objects belong to one class. Therefore, clustering should only be applied when the following condition is true: There are at least 2 objects in the scene and the difference of the areas of the smallest and the largest object is higher than a certain threshold $t_{minDiscriminationDifference}$. In practice, the absolute threshold approach resulted in better classification scores. Therefore I used the threshold approach in the experiments, although depending on the application, the clustering approach should be preferred.

The extraction of a semantic material label from the visual object descriptors is not a trivial task. While with the help of other sensory-motor perceptions of the textural surface by grasping an object the task would be easier, for the scope of this project an implementation based entirely on the visual texture has been implemented. After having tested various texture descriptors like the Haralick features, Gabor filters of the histogram of Local Binary Patterns, a solution based on the color histogram gave the best results. Based on the amount of pixels of the object that have very high or low brightness, one can calculate if an object is reflecting or not. As for the different experiments I used different materials, I will describe the used formulas in the respective section of the experiment.

Chapter 6

Experiments

In this chapter I will present the different experiments that were executed in order to evaluate the developed system. The two experiments will test the image segmentation on two VQA datasets. Firstly in the first experiment I will evaluate only the scene understanding system on the synthetic images of the CLEVR dataset [41] by answering questions on the scenes. In the second experiment, basic shape objects in a simple environment will be used to create scenes similar to the generated ones in the CLEVR dataset. On these scenes a NAO robot will be used to answer questions in the style of the CLEVR dataset. As a third example we will evaluate the potential of the emergent learning of concepts and grounding semantics in a simple grounded color naming game. For each of the experiments, I will discuss the setup, the evaluation and the obtained results.

6.1 Experiment description

In both experiments the task is to execute Visual Question Answering on environments consisting of 3 to 10 basic three dimensional geometric objects. These are in the following shapes: Cube, sphere or cylinder. They are also colored with a single color of a list of colors: Red, blue, green, yellow, purple, brown, cyan or gray and can of the material rubber or metal. The rubber material is very uniform and does not have any texture, while metal is reflecting its environment and light sources.

Each image has a set of questions which only have one unique answer.The questions test various aspects of visual reasoning including attribute identification, counting, comparison, spatial relationships and logical operations.

This is the general concept for both of the experiments executed in this thesis. However, a few changes exist between the two experiments which will be explained in their corresponding sections.

In both of the experiments, the general object descriptors returned by the implemented image segmentation are used to map to the semantic labels of the datasets. While in further advances of this project this linking should be learned incrementally by playing language games (see 2.2.2), for the purpose of testing the image segmentation and visual reasoning capabilities of the implemented system, the linking is done manually here. I explained the shared linking for both experiments in section 5.4, and the links that are only necessary in one of the experiments in the section of each experiment.

6.2 Evaluation methodology

The evaluation will be the same for both of the experiments. Since the focus of this thesis is on the developed image segmentation, I will test this part with greater detail than the full VQA system and the natural language part. The goal of the evaluation is to see how well the image segmentation system is detecting the objects in the test scenes. This can be done in various ways and will be split into different sub metrics to demonstrate how the different parts of the segmentation perform.

6.2.1 Object detection

To evaluate the quality of the object detection, which is to test if a segment in the scene has been correctly identified as an object and at the same position as the object in the ground truth scene, the following metrics will be used:

1. P: Precision: $\frac{TP}{TP+FP}$
2. R: Recall: $\frac{TP}{TP+FN}$
3. Percentage of scenes where all ground truth objects have been found
4. Percentage of scenes where all objects have been detected correctly and no false positives were detected
5. $\overline{d}$: Average error of position of the objects
6. $\overline{FP}$: Average false positives per scene

A detected object is counted as TP, if the detected center point is less than 20px far of the ground truth center point of the object. If multiple objects are detected for one ground truth object, only the closest one is counted as TP, while the others are counted as FP.

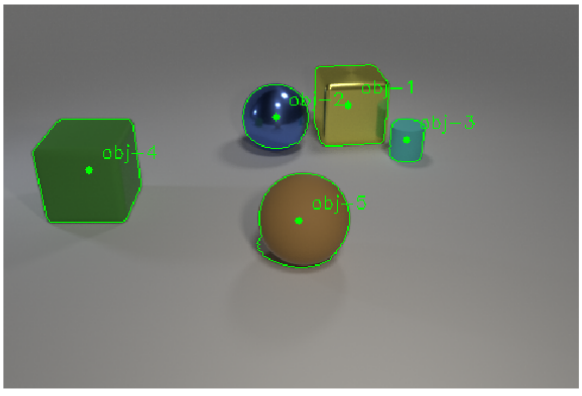


FIGURE 6.1: Example scene in which all objects have been correctly detected. Detected objects are outline by a green border and annotated with a incremental object id.

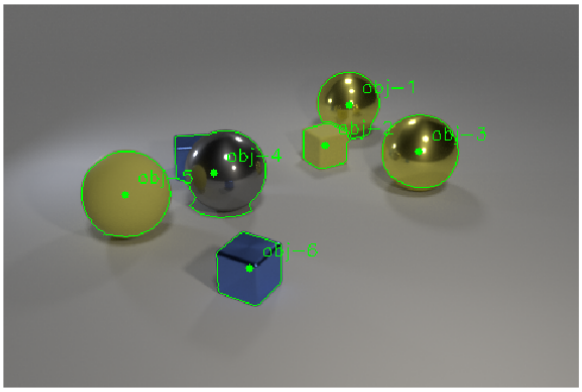


FIGURE 6.2: Examples for a scene with missing detected objects. The purple cylinder and the gray objects to the right are detected incorrectly. While *obj-2* is counted as correctly detected, *obj-2* is detected incorrectly since there is no corresponding object in the ground truth. *obj-3* is also detected incorrectly since its position is not close enough to the ground truth position of the gray sphere, while *obj-6* is counted as correctly detected but because of the incorrectly detected boundaries is not described correctly.

6.2.2 Object description

The correctly detected objects are described with discrete classes for their size (small or large), their color, their shape and their material (see 6.1). The evaluation of the predicted object descriptions will be performed with the following metrics:

1. Weighted precision and recall for entire classification
2. $DS_{correct}$: Percentage of scenes where all objects have been described correctly
3. Weighted precision for Color, Size, Shape and Material classification respectively
4. Weighted recall for Color, Size, Shape and Material classification respectively

It needs to be noted that these metrics can be only calculated on the objects that were correctly detected. Therefore a 100% of correctly described objects per scene does not imply that the scene was described 100% correct, since there might be objects that have not been detected and thus can not be described. However, these metrics were designed to evaluate the description algorithms of the image segmentation system and not the entire scene segmentation.

Finally I will also measure how many scenes have been segmented and detected entirely correct. That means, all ground truth objects have been detected correctly and were described correctly by all attributes, while no extra objects were detected.

6.2.3 Visual question answering evaluation

Since the implementation of this project was primarily in the image understanding part of the VQA system, I will test the overall VQA task only by one metric: how

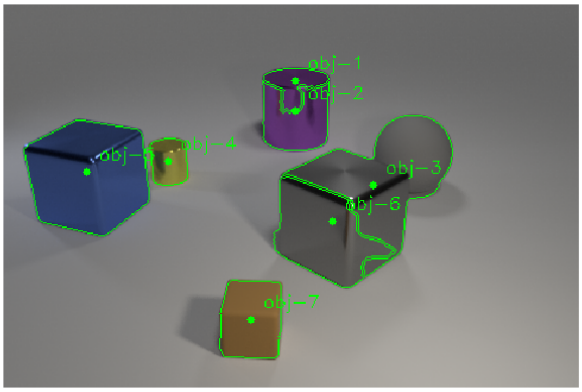


FIGURE 6.3: Example for a scene with incorrectly detected objects. The blue metallic cube behind the gray sphere labeled as *obj-4* is not detected as an individual object. However, *obj-4* is detected correctly, while its boundaries and descriptions might not be evaluated correctly. Therefore 6 out of 7 objects have been detected correctly and there are no objects detected that do not exist.

many of the questions have been answered correctly.

6.3 Experiment 1: Synthetic images

In the first experiment the performance of the image segmentation and the full VQA system will tested on the validation set of the CLEVR dataset. The CLEVR dataset is a VQA for testing compositional language and elementary visual reasoning dataset that was developed to overcome typical biases in datasets by being entirely generated. The images are rendered 3D-scenes with basic objects as described earlier. The questions are also generated by functional programs that can be used to construct question sentences. Although everybody could use the generators to create as many images and corresponding questions, the developers of the CLEVR dataset also provide training, validation and test splits of the data. The training data contains 70,000 images and 699,989 questions, the validation data contains 15,000 images and 149,991 questions and the test data contains 15,000 images and 14,988 questions. The test split does not contain the ground truth scenes. Since the image segmentation developed in this thesis is not intended to be trained and the ground truth for scenes only exists for the validation set and the training set, I will just use the validation set for the evaluation. Therefore the system will be tested on 15,000 images with a total of 94,302 objects.

Although the dataset is less biased by the creators and due to its generative, random origin perfectly balanced, it also has the disadvantage of being very predictable. Therefore, analyzing the building blocks - the basic geometric solids -, the typical scene organization and the questions, one can easily fit his/her implementation to the characteristics of the data set. Systems that are able to be achieve a close to optimal performance on segmenting the scenes and answering the VQA questions are likely reverse-engineering the generators of the data set.

Furthermore, the images in the dataset are almost completely noiseless and always have the same lighting. In real world scenarios, as we want to test with the NAO robot, one can expect the light conditions to change frequently. Since the camera of the NAO robot does not have a shutter and is therefore very light sensitive, a high amount of noise can be expected. The second experiment will evaluate the implemented system under more realistic conditions.

6.3.1 Object description

In this experiment, the material of the objects is either metal or rubber. To classify the detected objects by their object descriptors, the amount of white and black pixels is used:

$$L_{object}^{material} = \begin{cases} metal, & \text{if } p_{blacks} > 0.005 \vee p_{whites} > 0.005 \vee \sum_{k=1}^{3} M_{value}^{sorted}(k) < 0.5 \\ rubber, & \text{otherwise} \end{cases} \tag{6.1}$$

6.3.2 Results

In the following tables the results on the CLEVR dataset for object detection and object description are displayed. In table 6.1 the evaluation of the object detection can be seen: The tested system is able to detect objects with a good precision and high recall. Nevertheless, because of the high average of more than 1 object that is detected in addition to the objects in the scene, the percentage of scenes with correct object detection is quite low. The question if a high precision or a high recall is favorable is not trivial to answer in respect to question answering, since it depends on the question, as for example "Is there a ..." questions, testing the existence of objects, are less affected by false positive detections than others.

	Precision	Recall	All objects found	Correct scenes	$\overline{d}$	$\overline{FP}$
MS-WS	0.839	0.938	0.703	0.307	2.47px	1.16

TABLE 6.1: Experiment 1: Evaluation of the object detection method on the 15,000 scenes and 94,302 objects of the validation set.

The object description evaluation metrics are displayed in table 6.2. While color,size and material have very high precision and recall scores and shape has been classified rather badly. Interestingly, while in general the description for color and size is very accurate, in more than 30% of the scenes at least one object is described incorrectly for these attributes. Similar in shape and material, in only 17.1% and 42.1% of the scenes all objects are described correctly. This leads to an overall poor score of 11.1% of scenes, where all objects where classified correctly.

Observing the classification per class for colors, it can be seen that the system classified all colors with a f1-score higher than 0.9. The size of the objects is classified with a similar high f1-score on each of the classes. Investigating a few misclassified examples it can be assumed that misclassification on size is mostly due to the occlusion of objects and therefore incorrect shape estimation. This also affects heavily the described shapes of the objects, where more diverse scores for each class can be seen: Overall, the labeling of cylinders was the most difficult task for the system.

	Precision	Recall	$DS_{correct}$
Color	0.985	0.984	0.65
Size	0.970	0.969	0.598
Shape	0.833	0.735	0.171
Material	0.907	0.907	0.421
Overall	0.748	0.651	0.111

TABLE 6.2: Experiment 1: Evaluation of object description metrics for each attribute and overall. $DS_{correct}$ describes the percentage of scenes that were described correctly.

Cubes were described with high precision, while the sphere classification labeled the objects with high recall with low precision. The material was classified with a high precision and a high recall.

Summary and VQA evaluation

In summary, this evaluation shows that while the object detection system exposed quite good scores, the labeling of semantic classes by the computed object descriptors did not perform very well. The percentage of scenes in which all objects were detected and described correctly is only 6.25%.

When using the detect and described objects in the end-to-end VQA system and evaluating it on the 149,491 questions, 42% of the questions are answered correctly. Since evaluating the reason component of the VQA system on the ground truth scenes answers all questions correctly (100%), this means that the reasoning system is able to answer many questions even if the scene was not segmented entirely correct.

	f1-score	precision	recall	support
blue	0.964	0.934	0.997	11693
green	0.962	0.999	0.928	11518
red	0.994	0.999	0.989	11816
brown	0.979	0.972	0.986	11206
yellow	0.995	0.992	0.997	11219
purple	0.998	0.999	0.997	11714
cyan	0.998	0.999	0.998	11496
gray	0.982	0.985	0.980	10707
weighted avg	0.984	0.985	0.984	91369

TABLE 6.3: Experiment 1: Classification report for the color descriptor

	f1-score	precision	recall	support
small	0.971	0.946	0.998	48325
large	0.966	0.998	0.936	43044
weighted avg	0.969	0.970	0.969	91369

TABLE 6.4: Experiment 1: Classification report for the size descriptor

	f1-score	precision	recall	support
cube	0.709	0.968	0.559	29668
sphere	0.786	0.966	0.662	31138
cylinder	0.717	0.566	0.979	30563
weighted avg	0.738	0.833	0.735	91369

TABLE 6.5: Experiment 1: Classification report for the shape descriptor

	f1-score	precision	recall	support
metal	0.908	0.897	0.920	45991
rubber	0.905	0.917	0.893	45378
weighted avg	0.907	0.907	0.907	91369

TABLE 6.6: Experiment 1: Classification report for the material descriptor

6.4 VQA Experiment 2: Real images

For the second experiment the system will be tested on images captured by the camera of the NAO robot. Since setting up the scenes and annotating the images is a time consuming process, this dataset is much smaller: It contains 76 scenes with 760 questions and 345 objects. The goal of this experiment is therefore to test the visual reasoning capabilities of the system in a slightly more realistic environment. While still using basic objects and simple questions, the images are more varying in light conditions and the objects are more varying and not perfect.

6.4.1 Setup

As in the first experiment, the images have a uniform background and contain basic geometric solids. The background is a white plane taped to the floor and the wall. The geometric solids are made of plastic and paper. In comparison to the CLEVR dataset, the sizes of the object are varying more. Cylinders have varying radius and height while cubes and spheres have more than two different sizes. Furthermore, the pictures were captures in different light conditions, during different times of the day and with different artificial light sources. Figure 6.4 shows the robot and a sample scene with some of the objects.

Contrary to the first experiment, the images are annotated manually. By drawing a bounding box around each object and calculating the box center the 2D position in the image is annotated, shape, color, size and material are annotated manually. The dataset is published for further research.

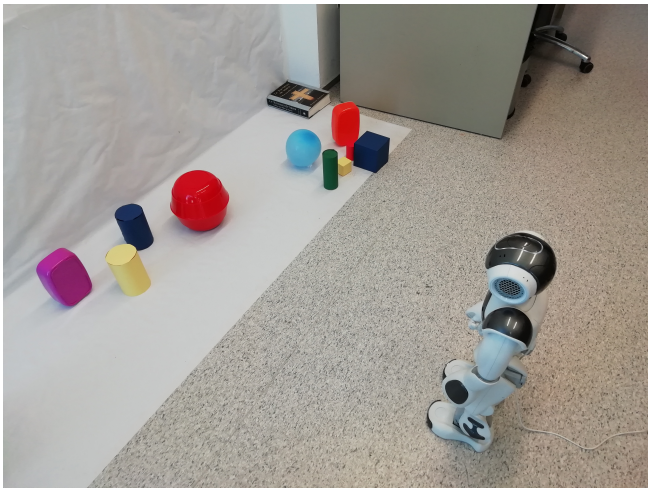


FIGURE 6.4: Experiment setup for the second experiment. The robot is located one meter from the scene he is looking at. The wall and the floor are covered with a white plane.

Object description

In the semantic description of the objects there is only one significant difference: While in the CLEVR dataset the material of the objects is metal and rubber, in this experiment objects are made of plastic or paper. Since the visual properties of plastic

and metal (at least for the scope of this experiment) and the ones for paper and rubber are very similar, only fine adjustments had to be implemented: While the metal objects in the CLEVR object have a high level of reflection, even other objects in the scene are reflected heavily and usually seen visually as black parts, whereas in the plastic objects in the real images only the light sources are reflected. Therefore in the CLEVR experiment, both the amount of very bright and very dark parts of the objects was used to link to the semantic class, while in the real experiment only the white parts have been used:

$$L_{object}^{material} = \begin{cases} plastic, & \vee p_{whites} > 0.005 \vee \sum_{k=1}^{3} M_{value}^{sorted}(k) < 0.5 \\ paper, & \text{otherwise} \end{cases} \tag{6.2}$$

6.4.2 Results

In the evaluation of the object description of this experiment, the F1-Score, the precision and the recall for the overall object description cannot be calculated, as the ground truth does not contain examples for all possible combinations of attributes. Instead, the accuracy score will be used, which by definition does not include the number of false positives.

In the evaluation of the object detection in the second experiment (see table 6.7) it can be seen that the objects were detected with high recall, but a significantly lower precision. However, in 75% of the scenes, all objects were detected. Since the system often detected one object as two parts of the object, in average 1.57 objects were detected which are not in the ground truth. Therefore in only 18.4% of the scenes all objects were detected without detecting additional, unmatched objects.

The detected objects were labeled by the attributes color, size, shape and material with high to medium precision and recall, ranging from around 98% precision and recall for color and size to 79% and 83% in precision for shape and material respectively and 74% and 77% respectively in recall for shape and material. As stated earlier, due to the incompleteness in objects of each combination of attributes, overall we can only evaluate the accuracy. In 54% of the objects the attributes were classified correctly.

Looking a bit closer into the results of the color classification, one can see that some colors were labeled entirely correct (green, red, yellow), while others (purple, magenta) were classified with worse precision and recall. Due to lighting conditions, those might have been confounded several times. Labeling the size of the objects was overall done quite precisely with a high F1-score of 98.4 %. The system did perform better on labeling objects as large, than as small. In shape classification an interesting result can be seen: Whenever an object was labeled as a sphere, it was a correct classification, while only less than half of the spheres in the ground truth were actually labeled as such. This might also be due to the not perfect round shape of the sphere objects used in the experiment. In comparison to the experiment with the synthetic images of the CLEVR dataset, cubes were labeled as such with a much lower recall in this experiment. Ultimately the results of the classification on material expose that too many objects were falsely labeled as plastic although they should have been classified as paper. Therefore paper has a high precision, while plastic has high recall.

	Precision	Recall	All objects found	Correct scenes	$\overline{d}$	$\overline{FP}$
MS-WS	0.725	0.919	0.75	0.184	3.44px	1.57

TABLE 6.7: Experiment 2: Evaluation of the object detection method on the 76 scenes and 345 objects of the data set.

	Precision	Recall	Accuracy	$DS_{correct}$
Color	0.975	0.973	-	0.447
Size	0.979	0.976	-	0.697
Shape	0.787	0.742	-	0.250
Material	0.826	0.766	-	0.263
Overall	-	-	0.541	0.066

TABLE 6.8: Experiment 2: Evaluation of object description metrics for each attribute and overall. $DS_{correct}$ describes the percentage of scenes that were described correctly.

Summary and VQA evaluation

In summary, this evaluation shows that while the object detection system also in the second experiment with real images exposed quite good scores, while object detection more often detected one object as two or more parts. In object description we experienced a few differences, but overall the evaluation results are similar to the ones of the first experiments. In the classification reports for each attribute it can be seen that the created dataset is not balanced well, but as we are using weighted averages to evaluate overall precision and recall, this does not affect the results. The percentage of scenes in which all objects were detected and described correctly is 6.58%.

When running the full visual question answering task on the 760 generated questions of the data set, the answer is correct in 30% of the cases. As only 6.58% percent of the scenes were described correctly, this means that the reasoning component of the VQA system is often able to answer a question with the correct answer even if the scene was not segmented entirely correct. Further investigation of how incomplete or incorrectly segmented data affects the reason capabilities are not in the scope of this thesis, but will be examined in future research of the group.

	f1-score	precision	recall	support
blue	0.986	0.972	1.000	103
green	1.000	1.000	1.000	43
red	1.000	1.000	1.000	51
yellow	1.000	1.000	1.000	79
purple	0.615	1.000	0.444	9
magenta	0.875	0.848	0.903	31
weighted avg	0.972	0.976	0.975	316

TABLE 6.9: Experiment 2: Classification report for the color descriptor

	f1-score	precision	recall	support
small	0.965	0.932	1.000	69
large	0.990	1.000	0.980	247
weighted avg	0.984	0.985	0.984	316

TABLE 6.10: Experiment 2: Classification report for the size descriptor

	f1-score	precision	recall	support
cube	0.731	0.742	0.721	68
sphere	0.644	1.000	0.475	80
cylinder	0.795	0.712	0.899	168
weighted avg	0.743	0.792	0.753	316

TABLE 6.11: Experiment 2: Classification report for the shape descriptor

	f1-score	precision	recall	support
paper	0.818	0.969	0.707	222
plastic	0.718	0.578	0.947	94
weighted avg	0.788	0.853	0.778	316

TABLE 6.12: Experiment 2: Classification report for the material descriptor

Chapter 7

Conclusions and Future Work

7.1 Conclusions

In this thesis I presented the implementation of an end-to-end Visual Question Answering system for the use in robots that is able to answer different types of compositional questions based on environments consisting of basic geometric solids of different sizes, shapes, colors and materials. The focus of my implementation was the vision system of the VQA system. After having evaluated several traditional computer vision methods for the different stages of the image understanding process, I implemented a new object detection and description method that does not need to be trained, uses few parameters and is deterministic. Additionally I considered and evaluated different methods to ground semantic labels for the categories *color, size, shape* and *material* using the computed object descriptors of the vision system and implemented grounding functions for the two experiments executed to evaluate the systems performance. The first experiment was performed on the synthetic images of the CLEVR dataset [41]. For the second experiment the NAO robot was used to create a new dataset with similar scenes and questions as in the CLEVR dataset. It is annotated and can be used for further research.

On both experiments the objects were detected with good precision and recall: 83.9% and 93.8% respectively for the CLEVR dataset, 72.5% and 91.9% for the new dataset. The classification of the object descriptors for the detected objects in semantic classes was evaluated with high precision and recall scores (>90%) on size, color and material for the CLEVR dataset, while in the second experiment only size and color classification exposed high precision and recall. Overall, the percentage of scenes where all objects have been detected and described correctly and no additional objects have been detected is under 10% in both experiments. The reasons for this low metrics are two-fold: First, the shape classification is not robust enough. Second, in average more than one object is detected additionally on each scene: Observing a few examples, frequent additional objects are object shadows or parts of an object, so that an object is detected as two separate objects. The latter describes a problem that is only solvable with a high level of semantic domain knowledge. An alternative to adding more semantic domain knowledge can be using more of the opportunities of employing the system in mobile robots. Since the robot is able to move, the scene could be analyzed from different points of view, allowing the robot to extract more low-level descriptors.

When evaluating the entire VQA system, interesting results were obtained: 42 % and 30 % of the questions respectively for the CLEVR dataset and the new dataset were answered correctly. Since those values are significantly higher than the percentage

of entirely correct segmented scenes, this means that the reasoning part of the VQA system, which was developed by the collaborators of Artificial Intelligence Lab of the VUB Brussels, is able to answer some questions correctly without having entirely correct context information. By looking at some examples of the evaluation, this is partly because for many questions it does not affect the question if additional objects have been detected or if some objects are labeled incorrectly, as for example the question *What is the color of the large shiny sphere?* from the CLEVR dataset can be answered correctly if the vision system classified exactly one object of the scene as a *large shiny sphere* with the correct color. The other detected objects, as well as their semantic labels, do not influence the system of answering the question. Until now, no further investigation of the reasons for this has been performed.

7.1.1 Intuitive grounding

Comprehensive experiments with state-of-the-art object descriptors was performed in order to find the best object descriptors for the task. An interesting insight of the experiments was, that often the intuitive, simple way to link a semantic label to the attributes of an object results in a more robust solution. For material classification, the percentage of very dark and very light pixels was found most discriminative. My personal conclusion from the development done in this thesis is that the intuitive, simple solutions might often turn out to be useful and robust. This insight might very well be used in evolutionary learning of concepts in a way that agents should always use the most discriminative and simple descriptor to ground new concepts.

7.2 Future Work

7.2.1 Quantifying the discriminative capabilities of object descriptors

The future directions of work that could be done next in the implementation of the vision system is to quantify the discriminative capabilities of each of the implemented object descriptors. This could give general insights of which object descriptors should be preferred when implementing a similar system or letting the agent decide dynamically which object descriptor to use.

7.2.2 Interactive question answering

Using the developed vision system as the foundation, the system could be extended into a interactive question answering system, in which the robot is able to analyze the scene from different points of view or could ask clarifying questions to the person asking the question. This field would also open a wide range of opportunities to concept learning and adaption in an evolutionary way.

7.2.3 Emergent incremental grounding of semantics

The research and development done in this thesis have built the foundation for many further experiments. While the implemented system is entirely static and not learning at all, in next steps more dynamic components could be introduced. Especially

for the further development of emergent grounding of semantic concepts, a flexible implementation should make use of the most discriminative object descriptor when referring to a detected object. A first step towards the evolution of semantic concepts could be to incrementally learn semantic labels by playing simple language games on the objects used in the VQA dataset and then evaluate the attribute classification after a certain number of iterations of the language game.

7.2.4 Reasoning with incomplete information

As the experiments of the end-to-end VQA system showed, the reasoning module of the system was often able to answer a question even with incomplete or incorrect scene information. Further directions of research how well the reasoning system answers questions if the visual information is noisy, incomplete or incorrectly perceived. Especially a evaluation per category could give interesting insights.

Bibliography

[1] Radhakrishna Achanta et al. *Slic superpixels*. Tech. rep. 2010.

[2] Aishwarya Agrawal et al. "Don't just assume; look and answer: Overcoming priors for visual question answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4971–4980.

[3] Stanislaw Antol et al. "Vqa: Visual question answering". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2425–2433.

[4] Pablo Arbelaez et al. *Contour Detection and Hierarchical Image Segmentation*. Tech. rep. UCB/EECS-2010-17. EECS Department, University of California, Berkeley, 2010. URL: http://www2.eecs.berkeley.edu/Pubs/TechRpts/2010/EECS-2010-17.html.

[5] Ann Marie Barry. *Visual intelligence: Perception, image, and manipulation in visual communication*. SUNY Press, 1997.

[6] Michael Van den Bergh et al. "Seeds: Superpixels extracted via energy-driven sampling". In: *European conference on computer vision*. Springer. 2012, pp. 13–26.

[7] Serge Beucher and Fernand Meyer. "The morphological approach to segmentation: the watershed transformation". In: *Optical Engineering-New York-Marcel Dekker Incorporated-* 34 (1992), pp. 433–433.

[8] Jeffrey P Bigham et al. "VizWiz: nearly real-time answers to visual questions". In: *Proceedings of the 23nd annual ACM symposium on User interface software and technology*. ACM. 2010, pp. 333–342.

[9] Gunilla Borgefors. "Distance transformations in digital images". In: *Computer vision, graphics, and image processing* 34.3 (1986), pp. 344–371.

[10] G. Bradski. "The OpenCV Library". In: *Dr. Dobb's Journal of Software Tools* (2000).

[11] Antoni Buades, Bartomeu Coll, and J-M Morel. "A non-local algorithm for image denoising". In: *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*. Vol. 2. IEEE. 2005, pp. 60–65.

[12] John Canny. "A computational approach to edge detection". In: *Readings in computer vision*. Elsevier, 1987, pp. 184–203.

[13] Dr Philippe Cattin. "Image restoration: Introduction to signal and image processing". In: *MIAC, University of Basel. Retrieved* 11 (2013), p. 93.

[14] M Emre Celebi. "Improving the performance of k-means for color quantization". In: *Image and Vision Computing* 29.4 (2011), pp. 260–271.

[15] Antonin Chambolle. "An algorithm for total variation minimization and applications". In: *Journal of Mathematical imaging and vision* 20.1-2 (2004), pp. 89–97.

[16] Yung-Yu Chuang et al. "A bayesian approach to digital matting". In: *CVPR (2)*. 2001, pp. 264–271.

[17] Dorin Comaniciu and Peter Meer. "Mean shift: A robust approach toward feature space analysis". In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 5 (2002), pp. 603–619.

[18] Dorin Comaniciu and Peter Meer. "Mean shift analysis and applications". In: *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Vol. 2. IEEE. 1999, pp. 1197–1203.

[19] K Dabov et al. "Image denoising by sparse 3-D transform-domain collaborative filtering." In: *IEEE transactions on image processing: a publication of the IEEE Signal Processing Society* 16.8 (2007), p. 2080.

[20] Abhishek Das et al. "Embodied question answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*. 2018, pp. 2054–2063.

[21] Abhishek Das et al. "Learning cooperative visual dialog agents with deep reinforcement learning". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2951–2960.

[22] David H Douglas and Thomas K Peucker. "Algorithms for the reduction of the number of points required to represent a digitized line or its caricature". In: *Cartographica: the international journal for geographic information and geovisualization* 10.2 (1973), pp. 112–122.

[23] Vyvyan Evans. *Cognitive linguistics*. Edinburgh University Press, 2006.

[24] Mark Everingham et al. "The pascal visual object classes challenge: A retrospective". In: *International journal of computer vision* 111.1 (2015), pp. 98–136.

[25] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. "Open question answering over curated and extracted knowledge bases". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM. 2014, pp. 1156–1165.

[26] Anthony Fader, Luke Zettlemoyer, and Oren Etzioni. "Paraphrase-driven learning for open question answering". In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vol. 1. 2013, pp. 1608–1618.

[27] Hao Fang et al. "From captions to visual concepts and back". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 1473–1482.

[28] Pedro F Felzenszwalb and Daniel P Huttenlocher. "Efficient graph-based image segmentation". In: *International journal of computer vision* 59.2 (2004), pp. 167–181.

[29] JC Aparício Fernandes and JAB Campos Neves. "Angle invariance for distance measurements using a single camera". In: *2006 IEEE International Symposium on Industrial Electronics*. Vol. 1. IEEE. 2006, pp. 676–680.

[30] Itzhak Fogel and Dov Sagi. "Gabor filters as texture discriminator". In: *Biological cybernetics* 61.2 (1989), pp. 103–113.

[31] Akira Fukui et al. "Multimodal compact bilinear pooling for visual question answering and visual grounding". In: *arXiv preprint arXiv:1606.01847* (2016).

[32] Keinosuke Fukunaga and Larry Hostetler. "The estimation of the gradient of a density function, with applications in pattern recognition". In: *IEEE Transactions on information theory* 21.1 (1975), pp. 32–40.

[33] Daniel Gordon et al. "Iqa: Visual question answering in interactive environments". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2018, pp. 4089–4098.

[34] Peter Gorniak and Deb Roy. "Grounded semantic composition for visual scenes". In: *Journal of Artificial Intelligence Research* 21 (2004), pp. 429–470.

[35] Yash Goyal et al. "Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 6904–6913.

[36] Robert M Haralick, Karthikeyan Shanmugam, et al. "Textural features for image classification". In: *IEEE Transactions on systems, man, and cybernetics* 6 (1973), pp. 610–621.

[37] Kaiming He et al. "Mask r-cnn". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2961–2969.

[38] Ming-Kuei Hu. "Visual pattern recognition by moment invariants". In: *IRE transactions on information theory* 8.2 (1962), pp. 179–187.

[39] Ronghang Hu et al. "Learning to reason: End-to-end module networks for visual question answering". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 804–813.

[40] Drew A Hudson and Christopher D Manning. "GQA: A New Dataset for Real-World Visual Reasoning and Compositional Question Answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2019, pp. 6700–6709.

[41] Justin Johnson et al. "CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning". In: *CVPR*. 2017.

[42] Justin Johnson et al. "Inferring and executing programs for visual reasoning". In: *Proceedings of the IEEE International Conference on Computer Vision*. 2017, pp. 2989–2998.

[43] Andrej Karpathy and Li Fei-Fei. "Deep visual-semantic alignments for generating image descriptions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3128–3137.

[44] Jin-Hwa Kim et al. "Multimodal residual learning for visual qa". In: *Advances in neural information processing systems*. 2016, pp. 361–369.

[45] Junkyung Kim, Matthew Ricci, and Thomas Serre. "Not-So-CLEVR: visual relations strain feedforward neural networks". In: (2018).

[46] Alex Levinshtein et al. "Turbopixels: Fast superpixels using geometric flows". In: *IEEE transactions on pattern analysis and machine intelligence* 31.12 (2009), pp. 2290–2297.

[47] Xiaodan Liang, Lisa Lee, and Eric P Xing. "Deep variation-structured reinforcement learning for visual relationship and attribute detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 848–857.

[48] Tsung-Yi Lin et al. "Microsoft coco: Common objects in context". In: *European conference on computer vision*. Springer. 2014, pp. 740–755.

[49] BPL Lo and SA Velastin. "Automatic congestion detection system for underground platforms". In: *Proceedings of 2001 International Symposium on Intelligent Multimedia, Video and Speech Processing. ISIMP 2001 (IEEE Cat. No. 01EX489)*. IEEE. 2001, pp. 158–161.

[50] David G Lowe et al. "Object recognition from local scale-invariant features." In: *iccv*. Vol. 99. 2. 1999, pp. 1150–1157.

[51] Jiasen Lu et al. "Hierarchical question-image co-attention for visual question answering". In: *Advances In Neural Information Processing Systems*. 2016, pp. 289–297.

[52] James MacQueen et al. "Some methods for classification and analysis of multivariate observations". In: *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. Vol. 1. 14. Oakland, CA, USA. 1967, pp. 281–297.

[53] David Marr and Ellen Hildreth. "Theory of edge detection". In: *Proceedings of the Royal Society of London. Series B. Biological Sciences* 207.1167 (1980), pp. 187–217.

[54] Nikolaos Mavridis and Deb Roy. "Grounded situation models for robots: Bridging language, perception, and action". In: *AAAI-05 workshop on modular construction of human-like intelligence*. 2005.

[55] Florica Mindru et al. "Moment invariants for recognition under changing viewpoint and illumination". In: *Computer Vision and Image Understanding* 94.1-3 (2004), pp. 3–27.

[56] Margaret Mitchell et al. "Midge: Generating image descriptions from computer vision detections". In: *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics. 2012, pp. 747–756.

[57] Anh Nguyen, Jason Yosinski, and Jeff Clune. "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 427–436.

[58] Hyeonwoo Noh and Bohyung Han. "Training recurrent answering units with joint loss minimization for vqa". In: *arXiv preprint arXiv:1606.03647* (2016).

[59] Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. "Multiresolution gray-scale and rotation invariant texture classification with local binary patterns". In: *IEEE Transactions on Pattern Analysis & Machine Intelligence* 7 (2002), pp. 971–987.

[60] Ioannis Pavlidis et al. "Urban surveillance systems: from the laboratory to the commercial world". In: *Proceedings of the IEEE* 89.10 (2001), pp. 1478–1497.

[61] Ethan Perez et al. "Film: Visual reasoning with a general conditioning layer". In: *Thirty-Second AAAI Conference on Artificial Intelligence*. 2018.

[62] Siddheswar Ray and Rose H Turi. "Determination of number of clusters in k-means clustering and application in colour image segmentation". In: *Proceedings of the 4th international conference on advances in pattern recognition and digital techniques*. Calcutta, India. 1999, pp. 137–143.

[63] Joseph Redmon and Ali Farhadi. "YOLOv3: An Incremental Improvement". In: *arXiv* (2018).

[64] Joseph Redmon et al. "You only look once: Unified, real-time object detection". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 779–788.

[65] Xiaofeng Ren and Jitendra Malik. "Learning a classification model for segmentation". In: *null*. IEEE. 2003, p. 10.

[66] Bernhard Riemann. "Grundlagen für eine allgemeine Theorie der Functionen einer veränderlichen complexen Grösse". PhD thesis. EA Huth, 1851.

[67] Leonid I Rudin, Stanley Osher, and Emad Fatemi. "Nonlinear total variation based noise removal algorithms". In: *Physica D: nonlinear phenomena* 60.1-4 (1992), pp. 259–268.

[68] Mark A Ruzon and Carlo Tomasi. "Alpha estimation in natural images". In: *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*. Vol. 1. IEEE. 2000, pp. 18–25.

[69] K. E. A. van de Sande, T. Gevers, and C. G. M. Snoek. "Evaluating Color Descriptors for Object and Scene Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 32.9 (2010), pp. 1582–1596. URL: https://ivi.fnwi.uva.nl/isis/publications/2010/vandeSandeTPAMI2010.

[70] Adam Santoro et al. "A simple neural network module for relational reasoning". In: *Advances in neural information processing systems*. 2017, pp. 4967–4976.

[71] Jianbo Shi and Jitendra Malik. "Normalized cuts and image segmentation". In: *Departmental Papers (CIS)* (2000), p. 107.

[72] Richard Socher et al. "Grounded compositional semantics for finding and describing images with sentences". In: *Transactions of the Association for Computational Linguistics* 2 (2014), pp. 207–218.
[73] Pierre Soille. *Morphological image analysis: principles and applications*. Springer Science & Business Media, 2013.
[74] Michael Spranger, Martin Loetzsch, and Luc Steels. "A perceptual system for language game experiments". In: *Language grounding in robots*. Springer, 2012, pp. 89–110.
[75] Luc Steels. *Design patterns in fluid construction grammar*. Vol. 11. John Benjamins Publishing, 2011.
[76] Luc Steels. *Experiments in cultural language evolution*. Vol. 3. John Benjamins Publishing, 2012.
[77] Luc Steels. "Introduction. Self-organization and selection in cultural language evolution". In: *Experiments in Cultural Language Evolution*. John Benjamins, 2012, pp. 1–37.
[78] Luc Steels. "The emergence of grammar in communicating autonomous robotic agents". In: (2000).
[79] Luc Steels. "The recruitment theory of language origins". In: *Emergence of communication and language*. Springer, 2007, pp. 129–150.
[80] Luc Steels. "The talking heads experiment". In: (2015).
[81] Satoshi Suzuki et al. "Topological structural analysis of digitized binary images by border following". In: *Computer vision, graphics, and image processing* 30.1 (1985), pp. 32–46.
[82] Wenbing Tao, Hai Jin, and Yimin Zhang. "Color image segmentation based on mean shift and normalized cuts". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)* 37.5 (2007), pp. 1382–1389.
[83] C-H Teh and Roland T. Chin. "On the detection of dominant points on digital curves". In: *IEEE Transactions on pattern analysis and machine intelligence* 11.8 (1989), pp. 859–872.
[84] Carlo Tomasi and Roberto Manduchi. "Bilateral filtering for gray and color images." In: *Iccv*. Vol. 98. 1. 1998, p. 2.
[85] Alain Trémeau and Philippe Colantoni. "Regions adjacency graph applied to color image segmentation". In: *IEEE Transactions on image processing* 9.4 (2000), pp. 735–744.
[86] Edward R Tufte and David Robins. *Visual explanations*. Graphics Cheshire, 1997.
[87] T Vlachos and AG Constantinides. "Graph—theoretical approach to colour picture segmentation and contour classification". In: *IEE Proceedings I (Communications, Speech and Vision)* 140.1 (1993), pp. 36–45.
[88] Jue Wang and Michael F Cohen. "An iterative optimization approach for unified image segmentation and matting". In: *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*. Vol. 2. 2005, pp. 936–943.
[89] Max Wertheimer. "Laws of organization in perceptual forms." In: (1938).
[90] Jason Weston et al. "Towards ai-complete question answering: A set of prerequisite toy tasks". In: *arXiv preprint arXiv:1502.05698* (2015).
[91] Kelvin Xu et al. "Show, attend and tell: Neural image caption generation with visual attention". In: *International conference on machine learning*. 2015, pp. 2048–2057.
[92] Charles T Zahn. "Graph theoretical methods for detecting and describing gestalt clusters". In: *IEEE Trans. Comput.* 20.SLAC-PUB-0672-REV (1970), p. 68.

[93] Peng Zhang et al. "Yin and yang: Balancing and answering binary visual questions". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016, pp. 5014–5022.