Semantic Scene Analysis Through Visual Exploration And Learning

by

Doğan Patar

B.S., Electrical and Electronics Engineering, Bilkent University, 2014

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Department of Electrical and Electronics Engineering

Boğaziçi University

2019

ACKNOWLEDGEMENTS

I would like to express my gratitude to my supervisor Prof. Işıl Bozma for accepting me to ISL. I am thankful for her continuous support, guidance and advices through out my studies.

I would like to thank Prof. Yağmur Denizhan and Prof. Hakan Temeltaş for being a member of my thesis committee.

This work has been supported in part by TUBITAK EEEAG-115E380.

Also, I would like to express my sincerest thanks to all ISL members but especially Kadir Türksoy, Mustafa Mete, Serhat İşcan, Berkan Höke, Meriç Durukan and Kemal Bektaş for their support and assistance.

Lastly, I would like to thank my parents Kenan and Fatma Patar, my sister Damla Patar for their patience and support.

ABSTRACT

Semantic Scene Analysis Through Visual Exploration And Learning

This thesis is concerned with the construction of object maps and their usage by a mobile robot endowed with a RGB pan-tilt camera. Such a representation is an important part of semantic scene analysis. However, it is a challenging task as the robot needs to be capable of object detection, scene exploration, object recognition and object mapping. We propose a series of approaches that address these problems so that the robot can construct an object map completely on its own. First, a novel approach that enables the robot to achieve both object detection and scene exploration simultaneously is proposed. In this approach, camera movements are guided by the so-far discovered object candidates. In parallel, the robot generates object candidates by tracking segments and determining spatio-temporally coherent ones. Following, an approach in which the generated object candidates are used in learning and recognition of object categories. In this approach, the robot has an evolving long-term object categories memory where the knowledge of learned object categories is organized in a hierarchical structure with each terminal node corresponding to an object category. The robot evolves its long-term object categories memory on its own through accumulating the unrecognized object candidates in its working object candidates memory. It waits for its working object candidates memory to fill up and then determines the new object categories which are then added to its long-term object categories memory. As such, the robot is able to construct an egocentric map of objects around it. In the case of revisiting a scene, a validation method that is based on comparing the object map of the current scene with those previously constructed is presented. All the proposed approaches are experimentally evaluated using visual data obtained from a mobile robot.

ÖZET

Anlamsal Sahnenin Görsel Keşif ve Öğrenme Yoluyla Analizi

Bu tezde, robot üzerindeki yatay dikey açı hareketli kamera kullanılar sahnenin nesne haritasını oluşturulması ve kullanımı incelenmiştir. Oluşturulan nesne haritası sahneyi anlamsal olarak analiz edebilmek için önemlidir; ancak, bu problemin çözümü için birçok sorununda değerlendirilmesi gerekmektedir. Nesne tespiti, sahnenin keşfi, nesnelerin tanınması, sahne haritasının oluşturulması ve doğrulanması değerlendirilmesi gereken sorunlardır. Bu çalışmada robotun kendi kendine nesne haritalarını oluşturabileceği şekilde her bir problem için bir dizi yaklaşım ileri sürlmüştür. İlk olarak, nesne tespiti ve sahne keşfinin eş zamanlı yapılabildiği bir yöntem önerilmiştir. Bu yönteme göre robot keşfettiği nesnelere göre kafa hareketlerini kontrol edip etrafına bakmaktadır. Bu esnasında da bölütlerin uzay-zamansal takibini yaparak, tutarlı bölütleri nesne adayı olarak belirlenmektedir. Ayrıca, bu tespit edilen nesne adaylarının öğrenilmesi ve tanınması için bir yöntem daha önerilmiştir. Bu yöntem iki ana kısımdan oluşmaktadır. Bunlardan ilki robotun gelişen uzun süreli nesne kategori hafızasıdır. Bu kısımda öğrenilen nesne kategorileri en alt düğümde bulunacak şekilde hiyerarşik bir yapı oluşturulur. İkinci kısımda ise robot tanıyamadığı nesneleri kısa süreli hafızasında biriktirir. Belli bir miktara kadar biriktiğinde yeni kategoriler oluşturulup uzun süreli hafıza güncellenir. Son olarak nesnelerin yatay-dikey açı pozisyonları ve kategorilerine göre robot merkezli nesne haritası oluşturulmaktadır. Robotun bir sahneyi tekrar ziyaret etmesi durumunda hafızadan gelen ve oluşturduğu sahne karşılaştırılarak doğrulama yöntemi sunulmuştur. Her bir yaklaşım deneyler yapılmış ve ayrıntılı olarak değerlendirilmiştir.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

$a(C_i^l)$	Attributes of ith segment of lth frame
A^l	Incoming frames
$B(V)$	Children nodes of V
$c(C_i^l)$	Mean color of segment C_i^l
$C \in \tilde{\mathcal{C}}^k$	Object candidate
$C_i^l \in \mathcal{C}^l$	ith segment of lth instance frame
d	Dimension
$f = [f_1, f_2] \in \mathcal{F}$	Viewing direction
g_V	Classifier function
$\mathcal{G}$	Objects' map
$k \in \mathcal{K}$	Index set of movements
$l \in \mathcal{L}$	Index
M	Segment existence matrix
$M_1(\mathcal{X}, \mathcal{Y})$	Normalized mutual information
$\mathcal{N}$	Nodes in the objects' map
N_c	Number of new object categories
N_l	Number of segments in lth frame
N_p	Minimum number of members allowed in object category
$\mathcal{N}_w$	Buffer size of WOCM
o_c	Object candidate with visual descriptor
O	Object class
$\mathcal{O}$	Object category set
q	Level of hierarchical structure
S^2	Pan-Tilt Space
t^l	Instance at lth frame
V	Node of hierarchical structure
$\mathcal{V}^k$	Object candidates that has been looked at
$w(C)$	Weight function

$\mathcal{W}$	Accumulated object candidates with descriptors
$\mathcal{X}$	Ground truth labels
$\mathcal{Y}$	Clustering labels
$\alpha(C, C')$	Overlapping percentage of two segments
δ	Minimum Segment Size
ϵ	Neighborhood of viewing direction
ζ	Encode relative weights
λ	Segment matching cost
$\Lambda^{ll'}$	Cost matrix
$\mu(C_i^l)$	Centroid of segment C_i^l
$\nu(C_i^l)$	Size of segment C_i^l
σ	Smoothing factor
$\rho(C)$	Radius of enclosing circle
τ_c	Maximum matching threshold
τ_l	Threshold for segment coherency
τ_m	Merging Threshold
τ_r	Categorization threshold
τ_w	Window parameter
Υ	Validation score
φ_k	Weighted gaussian mixture model

LIST OF ACRONYMS/ABBREVIATIONS

CNN	Convolutional Neural Network
CPU	Central Processing Unit
ISL	Intelligent Systems Laboratory
LOCM	Long-term Objects' Categories Memory
ROS	Robot Operating System
SVM	Support Vector Machine
TSP	Traveling Salesman Problem
WOCM	Working Objects' Candidates Memory

1. INTRODUCTION

A scene is an environment which consists of objects that are distributed in a meaningful way. In order to execute complicated tasks such as object manipulation, human interaction, a robot needs be able to analyze the scene. Objects are integral parts of a scene. They can determine the type of a scene. For example, an empty room can be transformed to bedroom or living room depending on the objects it contains. Therefore, the first step is detecting objects in a scene. This requires three distinct problems to be solved: i) object discovery; ii) categorization and learning; iii) object mapping. There has been extensive work done in each area as will is explained in the sequel. In general, they have been treated separately. In fact, they are quite related. This thesis focuses on the problem of constructing an object map of a scene through addressing these problems in an integrated manner - namely categorization and learning uses the generated object candidates and object mapping uses the results of both.

The first problem that is considered is the discovery of objects - namely the object candidates corresponding to the major items in its surroundings. As the robot's field of view generally tends to be limited, it is not possible for it to discover all the objects around it. Rather, it needs to look around. For this, a method that considers the two problems together is proposed. As such, the robot looks around autonomously and simultaneously discovers the objects candidates in its surroundings. The robot exploits scenes through tracking segments of uniform color and sufficient size over the incoming appearances and deciding spatio-temporally coherent segments as object candidates. Moreover, it explores its surroundings through the camera movements that depend on the so far found object candidates.

Secondly, after object candidates are found they need to be learned and recognized. For this problem, a novel approach is introduced in which a robot can continually learn object classes from unlabeled object candidates. In this approach, the recognition system has two significant parts. First one is a long-term object classes memory

where the knowledge of learned object classes are organized in a hierarchical structure in which each terminal node corresponds to an object class. Second one is that the robot evolves its long-term object classes memory on its own in an incremental manner through accumulating the knowledge of unrecognized object candidates in its working object candidates memory. It waits for its working object candidates memory to fill up and then processes the accumulated knowledge by first determining the new object classes and then incorporating this knowledge into its long-term object classes memory.

Lastly, detected objects are mapped based on the category label as well as the viewing position as expressed by the camera's pan and tilt angles. The constructed maps are evaluated through matching them with those that have been previously proposed. For this, a matching method is proposed. This is a simple method in which matching is based on first identifying objects belonging to the same category and then determining how similar the objects in their surroundings are. The latter is obtained through comparing these objects depending on the quadrant they occupy with respect to the object that have been found to be identical.

1.1. Contribution

The contributions are summarized as follows:

- Scene exploration and object detection: With the proposed approach, robot explores scene and detects objects simultaneously in autonomous way with no prior information. This allows robot to increase its field of view and collect more information about the scene and its objects.
- Recognition: An unsupervised incremental method is proposed. With the proposed method, robot can learn from the object candidates by itself. There are two main parts of this method. In the first part recognition is handled and in the second part learning on the unrecognized objects are conducted. This way only the unrecognized objects are considered for learning.
- Scene Validation: A method is proposed for comparing scene maps. Since mapping is done on a pan-tilt space, result significantly depends on the position of

the robot. The proposed method allows to make a comparison.

1.2. Organization of Thesis

The organization of this thesis is as follows. In Chapter 2, the generation of object candidates through scene exploration is presented and experimentally evaluated. Unsupervised and continual learning of object categories is explained in Chapter 3 along with an extensive experimental evaluation. Objects' mapping and matching is done as explained in Chapter 4. The thesis concludes with a brief summary and future directions. The details of robot hardware and associated software are given in Appendix B.

2. GENERATION OF OBJECT CANDIDATES THROUGH SCENE EXPLORATION

The goal is to enable a robot to look around and discover the objects in its surroundings. Such a capability enables the robot to determine object candidates without any prior knowledge on appearances or categories [1,2]. Our focus is on having a robot to find object candidates that correspond to the descriptive scene contents. The determined object candidates can then be sent one-by-one as queries to an object recognition system.

The outline of the chapter is as follows: First, related literature is reviewed briefly in Section 2.1. Then, the general approach of the proposed method is introduced in Section 2.2. The formula that governs the scene exploration is explained in Section 2.3. Following these movements, the generation of the object candidates is detailed in Section 2.4. Finally, in this chapter on robot experimental results are discussed in Section 2.5.

2.1. Related Literature

Object candidates are primarily defined by blobs or segments that encode perceptually similar pixels in an image [3]. In vision science, they are referred to as proto-objects since findings suggest them as being obtained from early segmentation processes [4]. Thus, object discovery is also known as 'proto-object perception'or 'object proposal generation'within cognitive psychology and the computer vision communities [5–9].

Most approaches to object discovery work with single images and use the image by generating hundreds of object candidates per image as to ensure the coverage of all the relevant scene objects [1, 2]. The trade-off between computational manageability and high discovery quality is addressed by considering 'detection proposals' which decrease

the object candidates [10, 11]. Alternatively, salient regions such as those with color contrast used to determine object candidates [12–15].

In all this work, the object candidates are determined from a single image. However, typically, robots have a limited field of view cameras. Consequently, a single image will not contain all the object candidates and the camera needs to be moved to cover the scene. With single-image based approaches, as the process is repeated in each image, the same objects can be discovered repeatedly in the incoming images and causing unnecessary computations. This can be eased by using video input and using the temporal coherence of the visual stream. Furthermore, the reliability of object candidates is shown to improve when using multiple views [16]. In most work, the video input is externally provided [17, 18]. Yet, the robot should be able to decide how to explore image regions that have not been looked at so that it can exploit them as is the case in natural vision [19]. While this has been pursued in exploration by having the robot bodily move in order to see objects from different points of view [20–23] or area coverage [24, 25], it has not been considered for enabling the robot to simply look around (without any body movement) and find object candidates in the incoming video.

2.2. Generating Object Candidates: General Approach

The proposed approach is to discover objects through a loop of visual exploitation and exploration considering only camera motion. The goal is to have the robot determine all the object candidates around it by simply looking around and covering the scenes in its surrounding. For example, in a sample case from a cluttered lab scene, the robot determines 19 object candidates as it covers roughly 95° pan and 35° tilt field-of-view as shown in Figure 2.1. The advantage of the proposed approach is that while it provides for the next viewing direction naturally towards unattended regions, the generated object candidates turn out to be united across the incoming visual stream. Initial work on this topic has been presented in [26]. Here, the approach has been reformulated as follows: i) Spatio-temporal coherency checking is revised; and ii) Both the robot's dynamics and the spatial relations between the object candidates

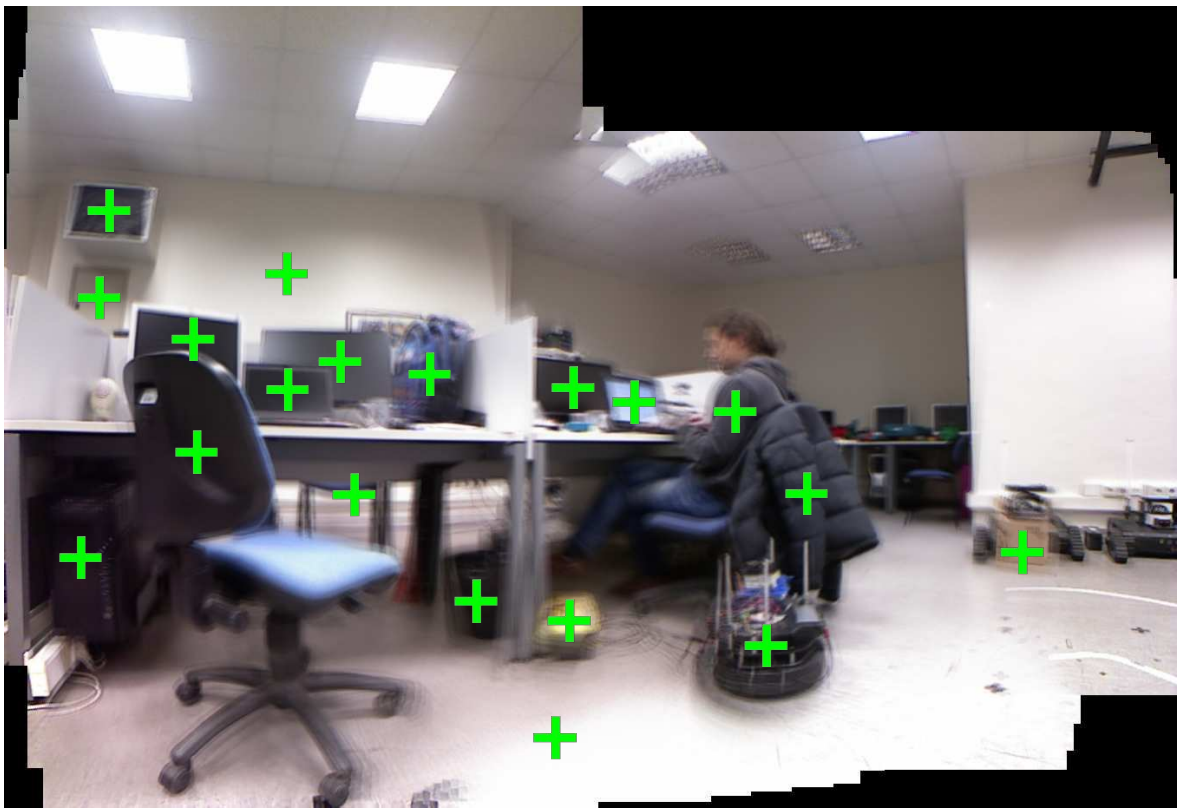


Figure 2.1. With the proposed approach, the robot couples visual exploitation and exploration. In this example, the robot has determined 19 object candidates (green crosses) after exploiting the incoming 136 frames that are acquired through 15 camera movements. The figure is obtained after stitching 15 frames with roughly 95° pan and 35° tilt coverage.

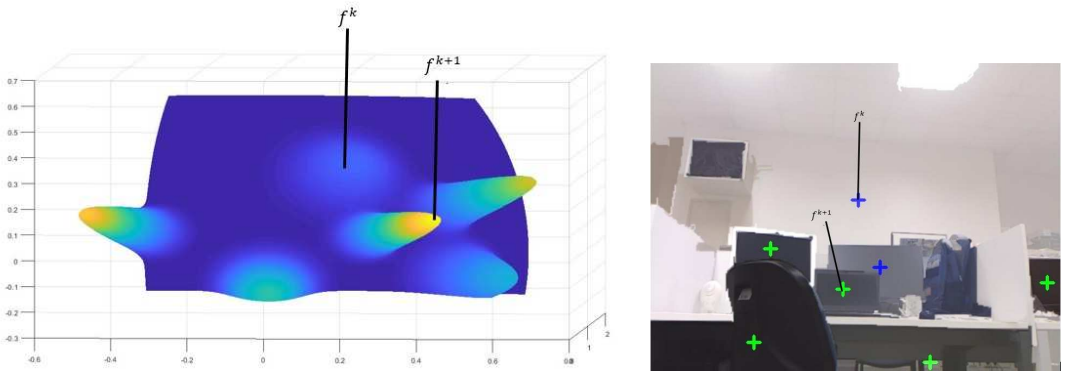
are expressed in the pan-tilt space. Our extensive experimental results with a robot operating indoors scenes varying in clutter as well as outdoors show that the robot is able to generate the object candidates in a non-repeating and consistent manner across the incoming images - in contrast to single-image methods. The parameters that are used in method are given in Table 2.1. Their usage are explained in the sequel.

2.3. Scene Exploration

We consider a robot with a pan-tilt monocular camera. The robot explores scene through a series of camera movements starting at times t_k, $k \in \mathcal{K}$ where $\mathcal{K}$ is the index set. Let $f^k \in \mathcal{F}$ its viewing direction at time t_k be defined as $f^k = \begin{bmatrix} f_1^k & f_2^k \end{bmatrix}^T$ with

Table 2.1. Parameters used in the generation of object candidates and scene exploration

Parameter	Definition
$\mathcal{F}$	Configuration Space
δ	Segmentation parameter minimum segment size
ϵ	Neighborhood of viewing direction
ζ	Relative weights of segment attributes
σ	Segmentation parameter smoothing factor
τ_c	Maximum segment matching threshold
τ_l	Threshold for segment coherency
τ_m	Segmentation parameter merging threshold
τ_w	Window parameter



(a) The function $\varphi_k(f)$ encodes 5 object candidates.

(b) The resulting movement starts at f^k and ends at f^{k+1}.

Figure 2.2. A sample movement: At onset, the robot is looking at an object candidate corresponding to the wall. With the movement, the robot ends up looking at an object candidate corresponding to a monitor.

pan $f_1^k \in S^1$ and tilt $f_2^k \in S^1$ angles where $S^1 \triangleq [0, 2\pi]$. Thus, $\mathcal{F} \subset S^2 \triangleq S^1 \times S^1$ is the configuration space of possible pan and tilt angles. The motion during the k-th movement is determined as the integral curve of a vector field that is automatically generated as the positive gradient of the function $\varphi_k : \mathcal{F} \to [0, 1]$:

$$\dot{f} = \nabla_f \varphi_k(f) \tag{2.1}$$

starting at $f(t_k) = f^k$. The advantage of our approach is that the robot does not have to explicitly compute the next viewing direction f^{k+1} prior to starting with its next movement. As such, the robot does not lose any time in its computation. Rather, the robot starts moving naturally and the next viewing direction is determined implicitly when the movement stops at time $t_{k+1} > t_k$ - namely when $\dot{f}(t_{k+1}) = 0$. Once this happens, the index is updated to $k+1$ and a new movement starts. These camera movements enable the robot look around in a continuously. The idea of defining the movement dynamics as such has been originally presented in [27]. Here, the formulation of φ_k is changed so that the resulting movement is guided by the so-far generated object candidates. In particular, it is now defined by a weighted Gaussian mixture model:

$$\varphi_k(f) = \sum_{C \in \tilde{\mathcal{C}}^k - \mathcal{V}^k} w(C) e^{-\frac{1}{2\rho(C)}(f-\mu(C))^T(f-\mu(C))} \tag{2.2}$$

The model is constructed with two considerations:
i) Camera movements should be towards object candidates $\tilde{\mathcal{C}}^k - \mathcal{V}^k$ at time t_k that have been previously generated, but not directly looked at. Here, $\tilde{\mathcal{C}}^k$ denotes the previously generated object candidates and $\mathcal{V}^k \subseteq \tilde{\mathcal{C}}^k$ are those that have been directly looked at. This construction is motivated by the fact that surrounding regions around such candidates are likely to contain the other object candidates that have not been yet discovered and hence the robot should direct its gaze towards these regions. Initially $\mathcal{V}^0 = \emptyset$. After each movement, the set $\mathcal{V}^k$ is iteratively computed as as $\mathcal{V}^k = \mathcal{V}^{k-1} \cup \mathcal{V}_k$ where the set $\mathcal{V}_k$ denotes the object candidates that are in the $\epsilon-$neighborhood of viewing direction f^k:

$$\mathcal{V}_k = \left\{ C_j^k \in \tilde{\mathcal{C}}^k \mid \exists f^m \in \mathcal{F}^k \text{ s.t. } f^m \in N_\epsilon(\mu(C_j^k)) \right\} \tag{2.3}$$

ii) Among those object candidates, there should be a bias towards moving to mid-sized ones close to the robot's horizon - motivated by human viewing behavior [28]. For this, the extent of attraction of each object candidate C depends on the radius $\rho(C) > 0$ of the enclosing disk. At the same time, it is weighted by $w(C) > 0$ that depends on the proximity of its centroid $\mu(C) \in \mathcal{F}$ to the horizon $f_2 = 0$ as well as its size $\nu(C) > 0$

in the images:

$$w(C) = \frac{ln(1 + 1/\nu(C))}{e^{|e_2^T \mu(C)|}}$$

where $e_2 = [0\,, 1]^t$ is the unit vector in the vertical direction. Thus, in contrast to a nearest frontier approach, camera movements consider a balance between proximity, size and closeness to the horizon. Looking around stops if the controller input is zero - namely $\dot{f}(t_k) = 0$. This happens if all the object candidates have been covered - namely $\tilde{\mathcal{C}}^k - \mathcal{V}^k = \emptyset$. This can also occur if the object candidates that have not been looked at are far away from its current viewing direction. In this case, the robot can simply move its camera directly to one of these object candidates and continue looking around in a similar manner. A sample movement is shown in Figure 2.2. Initially, the robot's viewing direction f^k is towards an object candidate corresponding to the wall. The constructed φ_k as shown in Figure 2.2(a) encodes 5 object candidates. The resulting movement ends up at f^{k+1} with the robot looking at the object candidate corresponding to a monitor. As such, the robot's field of view shifts towards unseen parts of the scene.

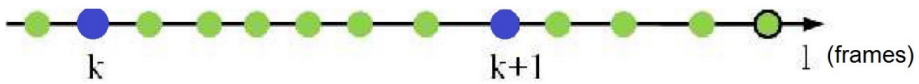


Figure 2.3. The robot acquires images (indicated by circles) throughout each camera movement (start and end of one as indicated by the two blue circles respectively) and updates the object candidates prior to starting the next movement.

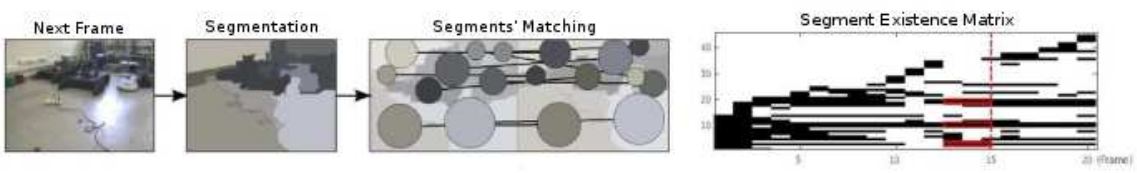


Figure 2.4. Object candidates are determined from tracking segments and determining spatio-temporally coherent ones.

2.4. Generating Object Candidates

The generation of object candidates is built upon a tracking method as shown in Figure 2.4. Thus, in comparison to single-shot methods, the generated object candidates turn out to be consolidated across the visual stream while inconsistent ones can be pruned through checking temporal coherence.

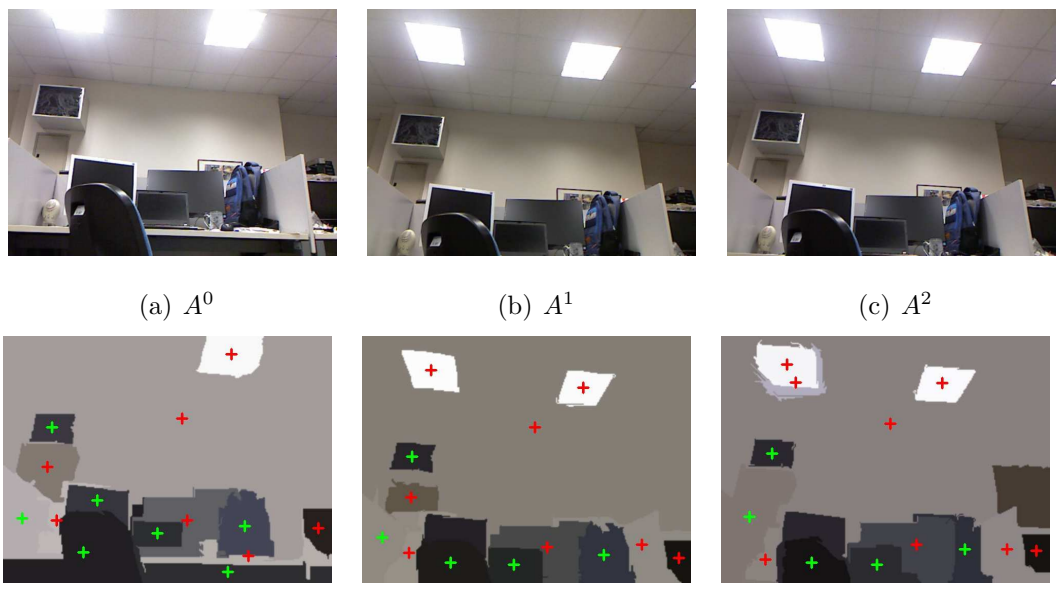
As the camera is moving, the robot tracks the segments across the incoming frames A^l where $t_k \leq t_l \leq t_{k+1}$. The robot can use any off-the-shelf segmentation method that is able to generate consistent segments with low computational complexity. Here, we use [29]. In this method, the number and size of the resulting segments are controlled by three parameters: The smoothing factor σ, minimum segment size δ affect the size of segments. The merging threshold τ_m affects their color uniformity. Their values are tuned so that the segments correspond to larger object candidates with relatively uniform color across them.

The resulting segments are then tracked across the incoming frames. Here, any off-the-shelf segments tracking algorithm could be utilized [30] pending that it should not assume a finite video stream (since the incoming video will be streamed in as long as the robot is looking around) and it should be computationally simple since it needs to be applied to each frame. Unfortunately, even with a good segmentation method, the tracking of segments across the incoming frames is not easy [18]. This is because as the robot is looking around through moving its pan-tilt camera, segmentation will not be consistent even between two consecutive images due to varying viewing direction and illumination effects. Some segments may be in one frame, but not in the next and there may be multiple segments in one frame that possibly match a single segment in the next. Our tracking method is based on minimizing an overall matching cost of segments' attributes. Such a method ensures a better tracking since all the segments are considered altogether while having low computational complexity. Let $\mathcal{C}^l = \left\{C_i^l \mid i = 1, \ldots, N_l\right\}$ be the set of N_l segments determined in image A^l. Then, the set of segments $\mathcal{C}^l$ associated with frame l is matched with those $\mathcal{C}^{l'}$, $l' < l$ previously

determined based on finding the minimum cost match using the cost matrix $\Lambda^{ll'}$.

$$\lambda_{ij}^{ll'} = \frac{\left\| a(C_i^l) - a(C_j^{l'}) \right\|_\zeta}{1 + \alpha(C_i^l, C_j^{l'})} \tag{2.4}$$

Its elements $\lambda_{ij}^{ll'}$ are defined as the weighted Manhattan distance between the segment attributes $a(C_i^l)$ and $a(C_j^{l'})$ of segment considering each pair of segments $C_i^l \in \mathcal{C}^l$ and $C_j^{l'} \in \mathcal{C}^{l'}$. Each segment $C_i^l \in \mathcal{C}^l$ is associated with three attributes as encoded by 3-dimensional vector $a(C_i^l)$: segment centroid $\mu(C_i^l)$, mean color $c(C_i^l)$ and size $\nu(C_i^l) > 0$. These attributes are selected as they are all easy to compute. The vector $\zeta = [\zeta_1, \zeta_2, \zeta_3]^T$ encodes their relative weights. Each element also encodes the overlapping percentage $\alpha(C_i^l, C_j^{l'}) \in [0, 1]$ of the two segments into account since a higher percentage increases the likelihood of a correct match. In order to have only low cost matches, only matches $\pi^{ll'}(i)$ with cost $\lambda_{i\pi^{ll'}(i)}^{ll'}$ less than the maximum matching threshold τ_c are considered to be valid matches.



(a) A^0 (b) A^1 (c) A^2

(d) $\mathcal{C}^0$ (e) $\mathcal{C}^1$ (f) $\mathcal{C}^2$

Figure 2.5. Generating object candidates. Top: A sample frame sequence. Bottom: Tracked segments that are spatio-temporally coherent (green cross) and those that are not (red cross).

After each camera movement and prior to the start of the next, the generated objects candidates $\tilde{\mathcal{C}}^k$ is updated as $\tilde{\mathcal{C}}^{k+1}$ through determining the spatio-temporally coherent segments in the frames obtained at times $t_k \leq t_l \leq t_k$ as shown in Figure 2.3. Spatio-temporal coherence is checked by determining whether they appear consistently in a pre-specified number of consecutive frames up to the current one. To this purpose, the matching results are maintained in a segment existence matrix $\mathbf{M}$. Its x-axis corresponds to the index set $\mathcal{L}$ and y-axis corresponds to the index set of all segments that have been tracked. For each frame A^l, $l \in \mathcal{L}$, the matrix entry $M_{li} = j$ if the segment C_j^l is matched to the segment C_i^m for some $l - \tau_w < m < l$. The window parameter τ_w designates how much of the history will be checked. As such, segments can be tracked even if they have disappeared for a short while - if this is less than τ_w. Its value is closely related to the frame rate. As it increases, so should τ_w. Unmatched segments are added as new y-axis entries in $\mathbf{M}$ so that the robot can determine the newly appearing object candidates. Segments with life spans exceeding a pre-defined threshold τ_l are determined to be spatio-temporally coherent. A larger τ_l enforces coherency more stringently. Thus, dynamic elements that appear only shortly are filtered out. A sample case is shown in Figure2.5. In the first frame as seen in Figure 2.5(d), there are 14 segments in total, 7 of which are spatio-temporally coherent (green crosses) while the rest (red crosses) are not. In the next frame of Figure 2.5(e), the robot finds 13 segments. It is observed that 5 of the spatio-temporally coherent segments from the previous frame continue to be so while 2 are no longer. Furthermore, 6 of the segments continue to be observed though they are not spatio-temporally coherent. There is 1 segment that has disappeared and 2 segments have newly appeared. In the third frame as shown in Figure 2.5(f), again 13 segments are found. The 5 spatio-temporally coherent segments remain. Out of the 8 segments that were observed, but not were spatio-temporally coherent, 7 continue to be observed with one segment disappearing and one segment appearing newly.

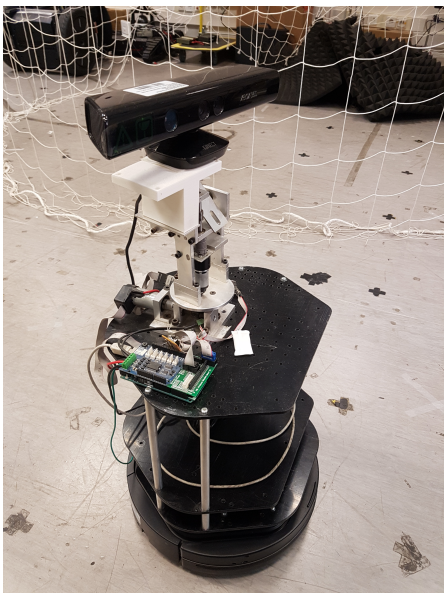


Figure 2.6. Kobuki turtlebot with pan-tilt camera

2.5. Experiments

We have experimented extensively with the proposed approach in various indoor and outdoor settings a robot endowed with a pan-tilt camera as shown in Figure 2.6 The pan-tilt mechanism has $\mathcal{F} = [-90°, 90°] \times [-30°, 30°]$. The camera has a $[-31°, 31°] \times [-23.3°, 23.3°]$ field of view and generates 480×360 images.

The segmentation parameters $\sigma = 0.5$, $\tau_m = 300$ and $\delta = 750$ are set manually as to have mid-size to larger segments. The attribute weights are $\zeta = \begin{bmatrix} 0.5 & 0.35 & 0.15 \end{bmatrix}$ so that centroid and area similarity are relatively weighted more in comparison to color similarity - as the latter is more sensitive to illumination. The parameter $\tau_c = 0.07$ is determined manually in order to get rid of matches with high costs. Segments are determined to be spatio-temporally coherent if they appear $\tau_l = 8$ in the last $\tau_w = 10$ frames. These parameters are set to enforce coherency stringently. In exploration, the parameter $\epsilon = 5°$. Our approach is computationally simple. Even though the robot does all its processing on a fairly simple Intel i5 CPU running at 1.7 GHz processor and without any code optimization, the robot is able to process around 2 frames per second. It is able to look around its surroundings roughly through 10-20 movements while processing roughly 100-200 frames. This means the robot is able to generate the

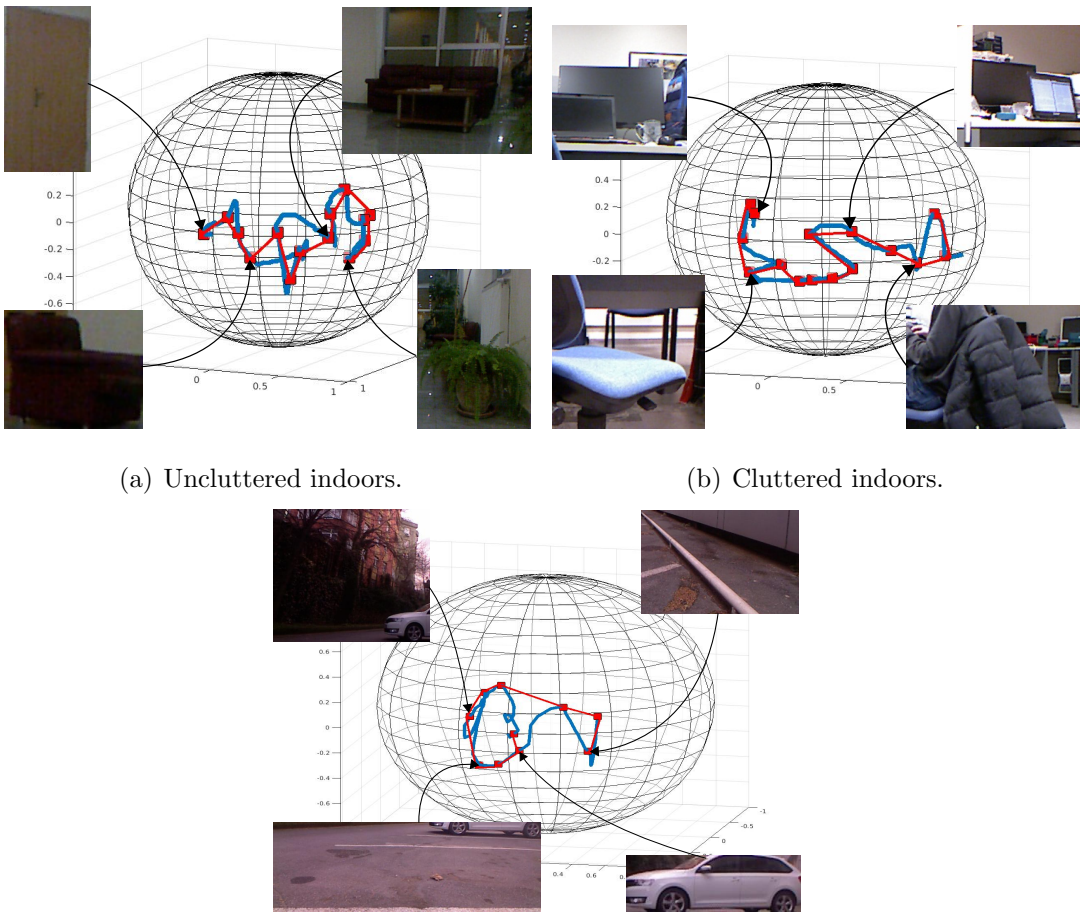
object candidates in its surroundings in between 1-2 minutes. The parameter values are also given in Table 2.2.

Table 2.2. Parameter values used in the generation of object candidates and scene exploration

Parameter	Value
$\mathcal{F}$	$[-90°, 90°] \times [-30°, 30°]$
δ	750
ϵ	$5°$
ζ	$[0.5, 0.35, 0.15]$
σ	0.5
τ_c	0.07
τ_l	8
τ_m	300
τ_w	10

In the first result we report, the robot looks around in a low clutter indoor setting. Through 14 movements along a path of length 242.86°, it covers a $(-5.2°, 63.5°)$ $\times$ $(-25.1°, 18.5°) \subset \mathcal{F}$ region as shown by the blue path in Figure 2.7(a). It has processed 149 frames and has discovered 11 object candidates out of the 17 ground truth (corresponding to 1 Table, 3 light fixtures, 3 couches, 2 windows, 3 doors, 3 plants, ceiling and ground) as determined externally by the human user. Those missed correspond to the two plants that are determined as one entity, the couch that is obstructed by the plants and the light fixtures due to their small size. We also compare this behavior with one if all object candidates were explicitly computed a priori. This is a case of traveling salesman problem (TSP) and and is solved using inexact algorithm based on genetic programming. The result as shown in red covers a path of length 130.08° Interestingly, with the proposed approach, the robot's path covers that of TSP to a greater extent while also containing new regions that are now looked at. Indeed we find this to be the case in all the experiments conducted.

The second experiment is from a cluttered indoor lab. The robot explores this setting in 15 movements covering $(-12.07°, 80.17°) \times (-18.20°, 16.51°) \subset \mathcal{F}$ as shown by the blue path in Figure 2.7(b). It is of length 238.69° as compared to 164.90° of inexact



(a) Uncluttered indoors.

(b) Cluttered indoors.

(c) Outdoors.

Figure 2.7. Exploration scanpaths (blue) vs TSP solution in $\mathcal{F}$. Sample object candidates that are discovered along the way are also given.

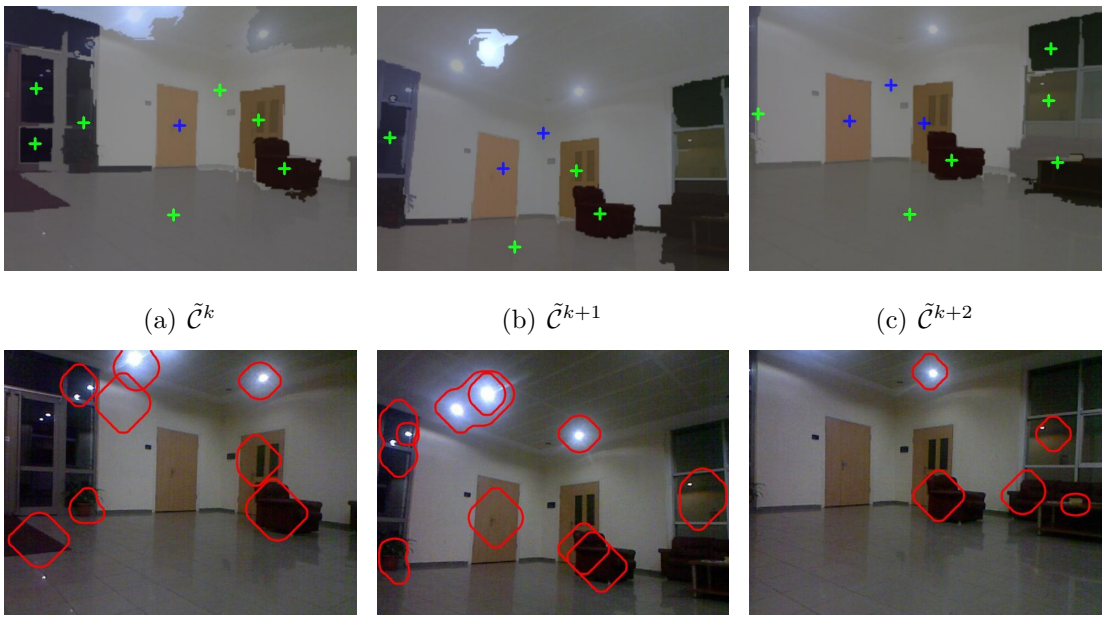
algorithm. In this case, the path is roughly 40% longer than optimal one. Throughout its movements, the robot processes 136 frames and finds 19 object candidates out of the 30 ground truth (those corresponding to wall, ground, ceiling, 7 monitors, 4 light fixtures, 2 chairs, 1 backpack, 1 coat, 1 painting, 1 box, 1 fuse box, 1 network cabinet, 1 human, 1 trash can, 1 tool box, 2 computers, 3 robots) as determined manually. As object candidates corresponding to monitors and light fixtures are either located at the boundaries of the field of view or are too small to be properly segmented, the robot is not able to discover them. Furthermore, there are some extraneous object candidates. This is attributed to the fact that due to factors such as size and illumination, the robot may generate two or three object candidates for an actual single object candidate.

The last experiment we report is done in an outdoor urban scene as shown by the blue path in Figure 2.7(c). The robot looks around its surroundings in 10 movements with a scanpath of length 217.03° as compared to 116.73° of TSP. Throughout the exploration, the robot processes 82 frames and is able to determine 7 object candidates out of 12 (3 buildings, a car, 5 trees, road, sky and grass) since trees and grass are merged to be one object candidate. This is attributed to the fact they cannot be discriminated using only perceptual cues.

Table 2.3. Comparative object candidates generation performance.

	Low clutter Indoors		Cluttered Indoors		Outdoors	
	[6]	Proposed	[6]	Proposed	[6]	Proposed
Input	Image	Video	Image	Video	Image	Video
Camera motion	External	Self	External	Self	External	Self
% Tracked object candidates	-	75	-	69	-	70
# New object candidates / Frame	8	1.77	10	2.43	10	1.85
Recall %	76	64	60	63	41.6	58
Precision %	64	73	54	61	23.3	66

Also a comparative study with a saliency based approach is conducted. We choose the method as presented in [6] since while most related work is similar in nature, its code is publicly available. Across these experiments, we note the following three advantages of the proposed approach as summarized in Table 2.3. First, while looking around is inherent to the proposed approach, this is not the case with the other method. Thus, the robot uses the incoming images - the other method does not have any incorporate any means of generating the camera movements. Second, while our approach enables the generation of a consolidated set of object candidates across the incoming visual stream, this is not the case with the other approach since the same objects are discovered repeatedly in consecutive frames. For example, object candidates as shown in Figure 2.9(d) are discovered again in the consecutive frame as shown in Figure 2.9(e) which are then discovered once again as shown in Figure 2.9(f). Thirdly, in contrast to the proposed approach, the consistency of generated object candidates is comparably lower - as an object candidate that is discovered in one image may be completely missed in the consecutive image while same object candidates are discovered

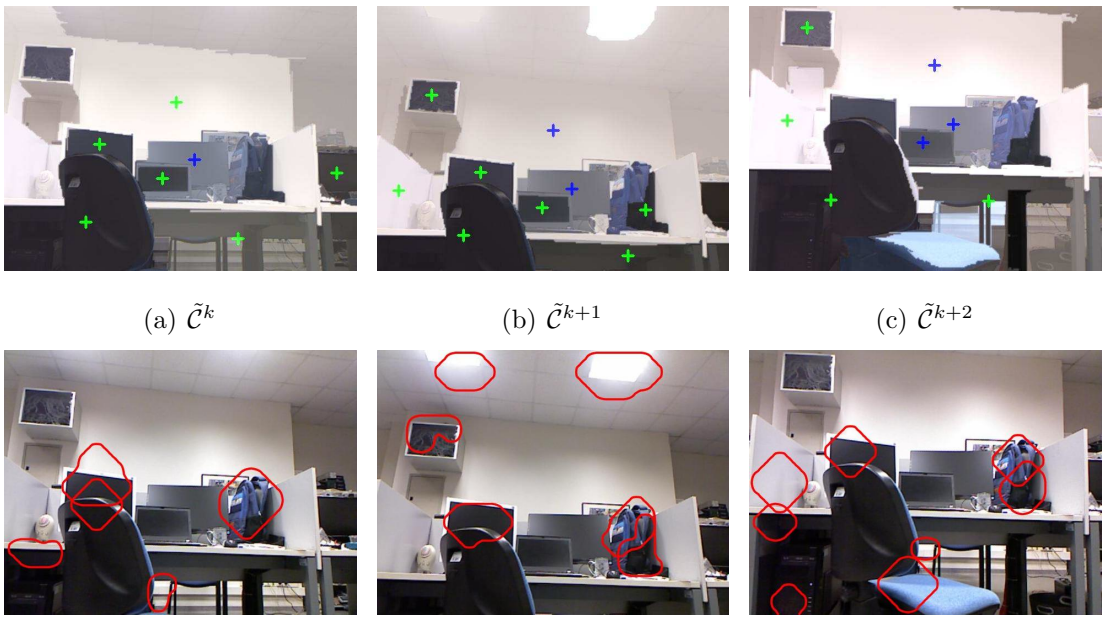


(a) $\tilde{\mathcal{C}}^k$ (b) $\tilde{\mathcal{C}}^{k+1}$ (c) $\tilde{\mathcal{C}}^{k+2}$

(d) $\tilde{\mathcal{C}}^k$ (e) $\tilde{\mathcal{C}}^{k+1}$ (f) $\tilde{\mathcal{C}}^{k+2}$

Figure 2.8. Generated object candidates in a low clutter indoor scene. Proposed approach (top row) vs [6] (bottom row). Note that in the latter, some object candidates are repeatedly or inconsistently determined across consecutive frames.

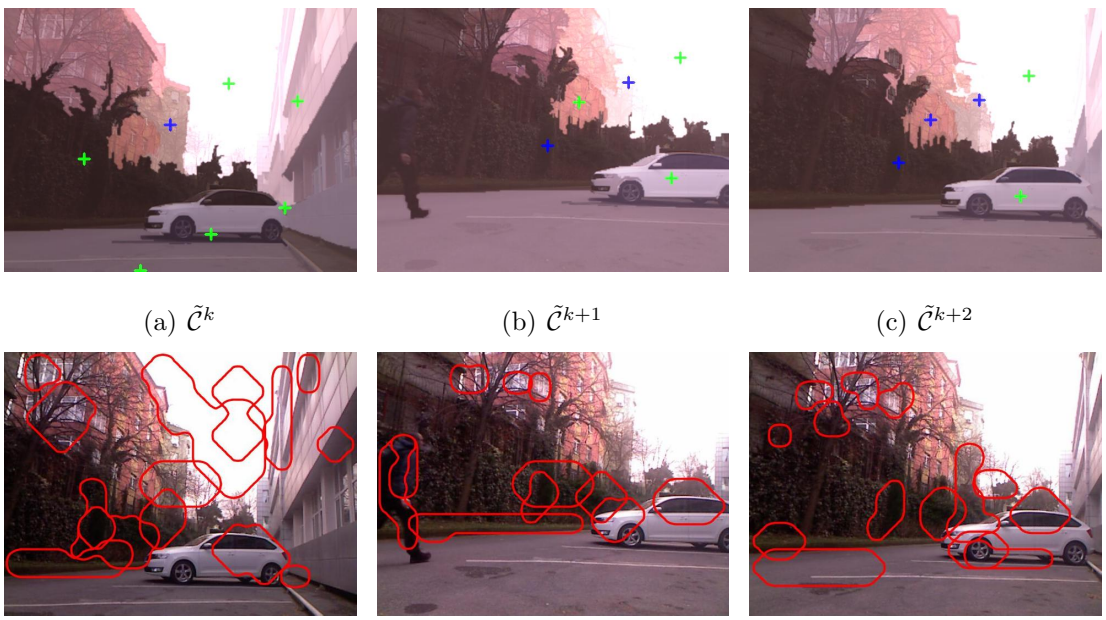
multiple times in a single image as observed in Figure 2.8, Figure 2.9 and Figure 2.10. For example, observe that one object candidate is discovered twice in Figure 2.9(e) while two object candidates are discovered twice in Figure 2.9(f). This results in their recall-precision rates to be different. Recall is defined by the ratio of generated object candidates while precision is defined by the ratio of correct object candidates. In both, ground truth is determined manually. In the low clutter scene, the proposed method has higher precision while recall is lower. In the cluttered indoor and outdoor settings, both rates are higher.



(a) $\tilde{\mathcal{C}}^k$ (b) $\tilde{\mathcal{C}}^{k+1}$ (c) $\tilde{\mathcal{C}}^{k+2}$

(d) $\tilde{\mathcal{C}}^k$ (e) $\tilde{\mathcal{C}}^{k+1}$ (f) $\tilde{\mathcal{C}}^{k+2}$

Figure 2.9. Generated object candidates in a cluttered indoor scene. Proposed approach (top row) vs [6] (bottom row).



(a) $\tilde{\mathcal{C}}^k$ (b) $\tilde{\mathcal{C}}^{k+1}$ (c) $\tilde{\mathcal{C}}^{k+2}$

(d) $\tilde{\mathcal{C}}^k$ (e) $\tilde{\mathcal{C}}^{k+1}$ (f) $\tilde{\mathcal{C}}^{k+2}$

Figure 2.10. Generated object candidates in an outdoor scene. Proposed approach (top row) vs [6] (bottom row).

3. LEARNING OF VISUAL OBJECT CATEGORIES

Many applications such as service and surveillance robotics require the robot to robustly categorize the objects that are in its surroundings as it is operating. This problem is challenging since the number of object categories is very large (≈30,000) with differing appearances [31]. Thus, it is not possible to program them before any deployment since that would require the learning of all object categories with offline training sets. Rather, the more fitting strategy is to make robots learn as necessary. In the sense, the robot's learned knowledge should evolve as necessary. Appearances play a significant role in learning. In most work, the robot needs to discover the object candidates - namely image segments without any prior information of their appearances or categories - in the incoming visual data [1, 2]. The object candidates are processed one-by-one in order to determine the corresponding object category.

In this chapter, a method is proposed to learn and categorize the incoming object candidates. First, related literature is reviewed briefly in Section 3.1. Then the general approach on the proposed method is introduced in Section 3.2. Afterwards, in Section 3.3, object categorization is explained. In Section 3.4 learning of the object categories is detailed. Finally, in Section 3.5, experimental results are discussed.

3.1. Related Literature

Categorization is a difficult problem because of the appearance and size variations amongst the objects as in typical work or home environments. Furthermore, in robotics, differing from computer vision benchmarks such as [32], these candidates do not just contain one object but have a central content with possibly some overlapping objects in the background [18]. Object candidates have been encoded using various representations ranging from using handcrafted descriptors to those obtained from clustering image patches [33,34] or from convolutional neural networks (CNNs) [35–37]. In most of studies, categorization has been often seen as a supervised problem. Most approaches including the state of the art deep convolutional neural network methods are

supervised methods and require the labeled training object candidates [38–44, 44–46]. While their recognition performance can be quite impressive, supervised teaching of all the objects is time-consuming and laborious.

As such, the challenge has been declared to be unsupervised visual object categorization [47]. The proposed unsupervised methods have been relatively much fewer in comparison to supervised methods. Most work are primarily focused on clustering - possibly coupled with the deep learning of representations [48–52]. In this framework, the concept of object hierarchies has been introduced. This has been motivated by the need to organize knowledge and by findings that even a manually constructed object hierarchy can improve categorization performance [53]. Object hierarchies are motivated by the observation that the organization should be based on appearance similarities. As such, the root represents the most general category while terminal nodes correspond to most specific categories. The automatic building of such a hierarchy has been proposed [36]. This idea has been extended to building a dynamic category hierarchy [54]. However, since both are built on the hierarchical Latent Dirichlet Allocation method, while the hierarchical structure is learned in an unsupervised manner, the number of levels is fixed and is provided externally. As such, the resulting hierarchies are limited when considering open-ended domains since the including the categories can possibly require new levels in the hierarchy.

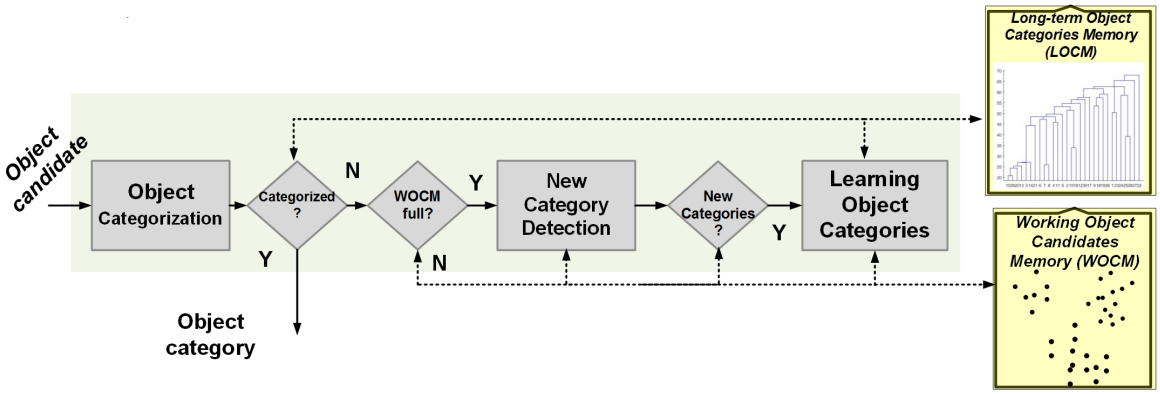


Figure 3.1. Life-long unsupervised learning of object categories.

3.2. Learning Object Categories: General Approach

We study this problem and propose a novel approach to the life-long and unsupervised learning of object categories from unlabeled images of object candidates. Our approach differs in two fundamental aspects. The first is related to the long-term object categories' memory (LOCM) in which the knowledge of learned object categories is stored in a hierarchical organization. Also, the number of levels and the structure of LOCM are completely free and evolve as necessary depending on the input object candidates. The second is related to how learning occurs. For this purpose, LOCM works in parallel with working object candidates memory (WOCM) that accumulates object candidates that have not been categorized. The clusters of object candidates that are determined without any supervision in the WOCM are then sent as the basis for the open-ended learning of object categories to the LOCM. Our final algorithm is fairly intuitive as shown in Figure 3.1. a The robot waits for the WOCM to fill up with unrecognized object candidates and then learns new object categories from the accumulated information. The learning process consists of first determining the new object categories followed by combining this information into its LOCM. This is repeated throughout its operation. Thus, the robot is able to evolve its LOCM in an incremental manner on its own throughout its operations. To summarize, the major contributions are:

(i) We propose a simple but effective approach to life-long and unsupervised learning of object categories from an unlabeled object candidates in which the long-term object categories memory evolves without any restrictions regarding the number of levels or its structure;

(ii) We formulate the learning of object categories as two separate, but coupled processes: new category detection in the WOCM followed by new category learning in the LOCM.

(iii) Our experimental results show the robot can learn meaningful object categories from unlabeled object candidates throughout its operations with high precision.

The proposed approach requires a set of parameters as described in Table 3.1. Their usage is detailed in the sequel.

Table 3.1. Parameters used in learning and categorization

Parameter	Values
N_p	Minimum number of members allowed in object category
$\mathcal{N}_w$	Buffer size of WOCM
τ_r	Categorization threshold

3.3. Object Categorization

The goal of object categorization is to relate each object candidate to the long-term objects categories memory (LOCM). From the Chapter 2, the robot is assumed to have an incoming stream of unlabeled images of object candidates C. Each object candidate C is assumed to be represented by a $d-$dimensional visual descriptor $o_c \in R^d$. The descriptors can be constructed using a various representations - as long as similar objects generate similar descriptors. In this work, two different descriptors are used. The first one is the bubble space representation due to demonstrated advantages such as nearby appearances generating similar descriptors, encoding both local sensory features and their relative S^2 geometry, incorporating any number of observations and being rotationally invariant [55]. The second descriptor is a descriptor obtained from f6 layer of a convolutional neural network Alexnet [56].

LOCM enables the robot to construct, store and retrieve the visual knowledge of object categories as indexed by the set $\mathcal{O}$ with cardinality $|\mathcal{O}|$. It has a hierarchical structure as defined by a sequence of partitions of the set $\mathcal{O}$. The partition at the top level is the whole set $\mathcal{O}$ while each of terminal nodes corresponds to a distinct object classes $O \in \mathcal{O}$. All other nodes correspond to particular clusters of $\mathcal{O}$. In particular, object classes belonging to each cluster can be viewed as sharing certain common attributes. As such, each node V except the root node is associated with a classifier function g_V. This function measures how likely an object candidate is to be associated with the object classes belonging to node V. Thus, the structure of LOCM

can be viewed encoding the perceptual hierarchy among different object classes. As such, it provides a basis for symbolic grounding of object categories. For example, they may be associated with the corresponding natural language labels.

Memory association is achieved through traversing down the LOCM hierarchy level-by-level and evaluating a sequence of g_V functions at the traversed nodes. At each level q, the robot finds a node V^q with the minimum cost function $g_{V^q}(o_c)$ considering the children nodes $B(V^{q-1})$ of the last node V^{q-1} reached while ensuring that the minimum cost is above a categorization cost threshold τ_r:

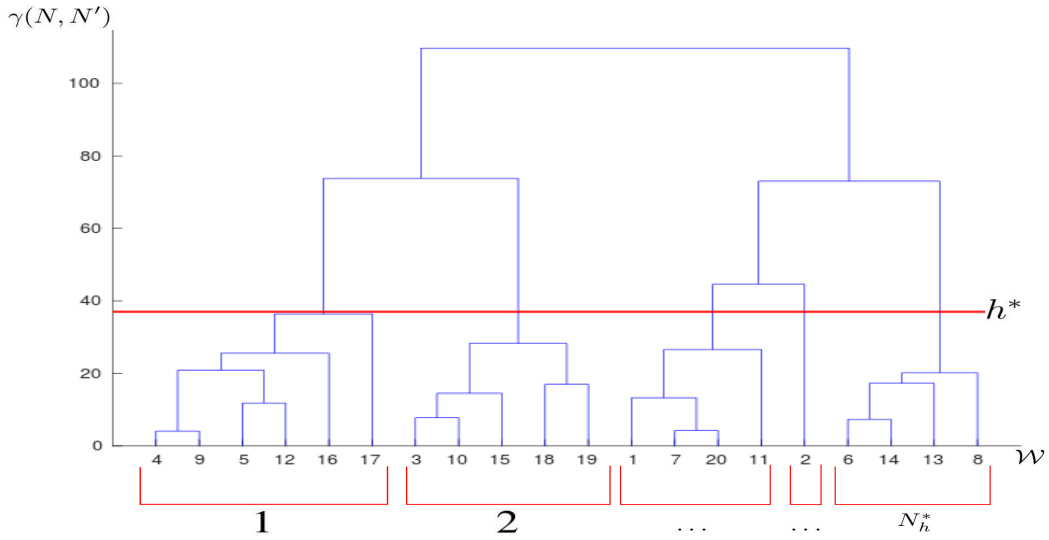
$$V^q \in \arg \min_{V' \in B(V^{q-1})} g_{V'}(o_c) \ \geq \tau_r \tag{3.1}$$

The cost function g_V is the one-class SVM classifier that is associated with node V. The categorization threshold τ_r determines the trade-off between recall and precision. As its value is decreased, while precision decreases, recall increases. On the other hand, as its value increases, recall decreases while precision increases. As such, it can be viewed as a confidence measure. In general, in order to minimize false positives, recall at 100% precision has been the key metric in performance assessment. In case there are multiple children nodes that pass this check, then among them, the node with minimum variation as measured by the norm of covariance matrix $\|Cov(V')\|$associated with the node V' is selected. This process is repeated until either one of the terminal nodes is reached or the condition of Equation 3.1 is not satisfied. In the former case, decisions are progressively combined to hierarchically refine the final decision so that the object candidate is recognized as belonging to the object category associated with the reached terminal node. In the latter case, no categorization decision can be made. This implies that the incoming object candidate o needs to be used in learning.

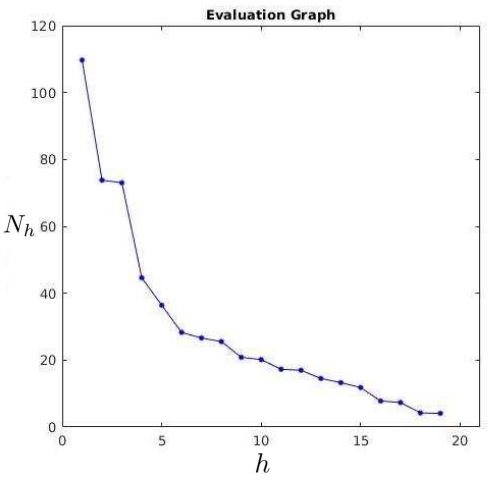
3.4. Learning Object Categories

In case the robot is not able to categorize an object candidate, it then accumulates this information in its working object candidates memory (WOCM) for the purpose of determining new object categories. Thus, WOCM acts as a small temporary buffer

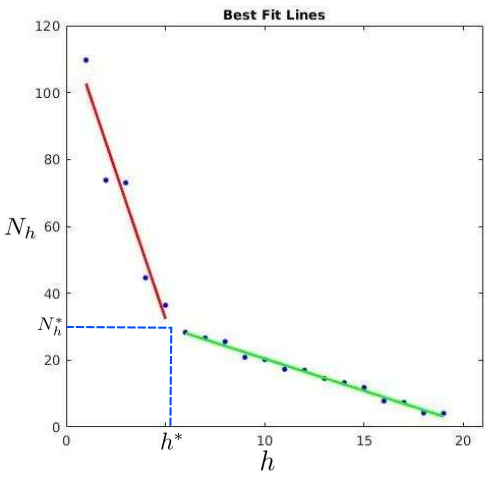
(of size $\mathcal{N}_w$) for the accumulated object candidates $\mathcal{W}$. Typically, the input object candidates will correspond to different objects so that the robot will not have the same object images. Throughout its operation, the robot waits for the WOCM to fill up and then processes the accumulated data. The processing of unrecognized object candidates consists of two stages: detecting new object categories and updating its LOCM.



(a) Hierarchical clustering in WOCM.



(b) h vs N_h



(c) L-method

Figure 3.2. The new object categories are determined by the partition of $\mathcal{W}$ with height h^*.

3.4.1. Detecting New Object Categories

The detection of new object object categories $\mathcal{O}_n$ is built upon clustering of accumulated object candidates $\mathcal{W}$ using Ward's method [57]. This is an agglomerative hierarchical clustering that considers within-cluster variance. Namely, for pair of clusters $\mathcal{D}(V)$ and $\mathcal{D}(V')$, this measure is defined as:

$$\gamma(V, V') = \sum_{q \in \mathcal{D}(V)} (q - \mu)^2 + \sum_{q \in \mathcal{D}(V')} (q - \mu')^2 - \sum_{q \epsilon \mathcal{D}(V) \cup \mathcal{D}(V')} (q - \nu)^2$$

where

$$\mu = \frac{1}{|\mathcal{D}(V)|} \sum_{q \in \mathcal{D}(V)} q, \quad \mu' = \frac{1}{|\mathcal{D}(V')|} \sum_{q \in \mathcal{D}(V')} q$$

and

$$\nu = \frac{1}{|\mathcal{D}(V) \cup \mathcal{D}(V')|} \sum_{q \in \mathcal{D}(V) \cup \mathcal{D}(V')} q$$

We prefer this method since it aims to minimize the total within-cluster variance. As such, the purity of each new category can be better ensured. A sample hierarchy is shown in Figure3.2(a). The time complexity of this method is $O(\mathcal{N}_w^2)$ [58].

The resulting hierarchy can be used as determining the number $N_c = |\mathcal{O}_n|$ of new object categories - as the robot does not know it a priori. This method is based on the observation that each fixed distance $h \in R^{\geq 0}$ corresponds to one different partition of $\mathcal{W}$ with cardinality N_h. Thus, the hierarchy defines a map between h and N_h as shown in Figure 3.2(b). The robot then fits two lines to the resulting map using L-method [59] and determines the pair (h^*, N_{h^*}) corresponding to the intersection of these two lines as shown in Figure 3.2(c). The new object categories are determined by the

partition of $\mathcal{W}$ with height h^*. The value N_{h^*} gives an upper bound for the number of new object categories. The L-method enables the robot to determine these categories correctly even if the robot is input multiple images of the same object candidate. In this case, the robot is likely to find one or few new categories depending on the appearance similarity of the incoming candidates. However, only those having at least N_p members are passed to the LOCM to be added as new object categories. The rest continue to remain in the WOCM. Let the number of newly determined object categories be $N_c = |\mathcal{O}_n|$ with $N_c \leq N_{h^*}$.

3.4.2. Updating LOCM

Next, the robot updates its LOCM as to combine the new determined object categories $\mathcal{O} = \mathcal{O} \cup \mathcal{O}_n$. To this purpose, first the hierarchy of the LOCM is updated and then the classifiers at the changed nodes of the hierarchy are relearned. The LOCM hierarchy is also updated using the newly determined object categories O_j, $j = 1, \ldots, N_c$ using the SLINK method [60]. This is also an agglomerative hierarchical clustering method, but with both storage and construction efficiency as well as the resulting hierarchy being order-invariant. Here, the mean descriptor of each new object category O_j is used to update the LOCM hierarchy. The hierarchical structure also provides means of associating with each node V except the root node with a discriminant function g_V for object class recognition [61–64]. This is based on one class support vector machine (one-SVM) [65]. Each g_V is learned using the object candidates associated with the node. When the tree hierarchy is updated, only the SVMs that are at the affected nodes are retrained. While the training data reduces going down the hierarchy, the data amount is sufficient even at the terminal nodes as long as each new object class has sufficient number of samples as accumulated in WOCM.

3.5. Experimental Results

The proposed approach has been evaluated on a robot that starts out having no knowledge of object categories in its LOCM. Although there are well established methodologies to evaluate learning systems, these are primarily for supervised ap-

Table 3.2. Object categories in mobile robot data set

Air-conditioner	Backpack	Black Panel	Bookshelf
Box	Building	Car	Ceiling
Central Heating	Chair	Computer Case	Couch
Curtain	Cylinder	Door	Drawer
Dressing Screen Frame	Floor	Fuse Box	Hanger
Jaguar Robot	Light Fixture	Monitor	Person
Phone	Plastic Bag	Poster	Road
Sidewalk	Switch Box	Table	Tree
Umbrella	Wall	Wardrobe	Water Fountain
White Board	Window		

proaches [34]. As they require the categories to be known, they do not do the simultaneous nature of categorization and learning. Thus, they are not well suited for evaluating open-ended unsupervised learning systems. As such, we follow a method that has been suggested for open-ended learning systems in which the interactions with the surrounding environment is emulated over a period of time [66]. To this purpose, we consider a data set that is generated by an approach explained in previous Chapter 2. This data set contains 14890 object candidates from 38 ground truth object categories that are commonly observed by the robot as shown in Table 3.2. Note that some objects such as wall, whiteboard and wardrobe are difficult to discriminate from each other based purely on appearance. Also, some input object candidates possibly contain multiple objects where there is one (smaller) object is at center dominating and others appear partially in the background. The robot is input these object candidates one at a time with random order and uses the proposed model to evolve its LOCM. The parameters are for the proposed method summarized in Table 3.1. If otherwise is not stated, the parameters are set as in Table 3.3.

Table 3.3. Parameter values in learning and categorization

Parameter	Values
N_p	20
$\mathcal{N}_w$	3000
τ_r	0

As the robot's LOCM is initially empty, we first study how well the robot is able to determine the new object categories in its WOCM. For this, we compute normalized mutual information $M_1(\mathcal{X}, \mathcal{Y})$ that is associated with the resulting new object categories [67]:

$$M_1(\mathcal{X}, \mathcal{Y}) = \frac{2 * I(\mathcal{X}, \mathcal{Y})}{H(\mathcal{X}) + H(\mathcal{Y})} \tag{3.2}$$

Here, $\mathcal{Y}$ represents the set of clustering labels. It is an external measure because we need the ground truth labels $\mathcal{X}$. As such, mutual information $I(\mathcal{X}, \mathcal{Y})$ and respective entropies $H(\mathcal{X})$ and $H(\mathcal{Y})$ of the sets $\mathcal{X}$ and $\mathcal{Y}$ are computed. We prefer M_1 since it is not biased toward a higher number of clusters. $0 \leq M_1(\mathcal{X}, \mathcal{Y}) \leq 1$ with higher values preferred. Next, we study the complete system - namely WOCM and LOCM operating in an integrated manner. The parameters are $\mathcal{N}_w = \{3000\}$, $N_p = 20$ and $\tau_r = \{0\}$. The robot processes 14890 object candidates. The encoding of the object candidates are known to be important for learning and categorization. For this, two alternative descriptors are considered for encoding the object candidates: bubble descriptors and Alexnet descriptors.

3.5.1. Bubble Descriptors

First, object candidates are encoded using bubble descriptors $o_c \in R^d$ with $d = 500$ [55]. These are hand-crafted descriptors that encode Cartesian and Non-Cartesian filter responses and hence do not need any any learning. The M_1 values for scenarios are shown in Table 3.4 using bubble descriptors. Here note that the number of categories N_h^* is self-determined. Interestingly, increasing WOCM buffer size results in a lower M_1. This suggests that as more object candidates accumulate, there is a higher chance of confusing different object categories. Through closer inspection, we attribute this to the fact there is an increased chance of different object candidates having similar objects appearing partially in the background. For comparison, we also evaluate compute M_1 if the number of categories was set as $N_{h^*} = 38$. A similar trend is observed in this case as well. Furthermore, we also consider K-means clustering with the number of clusters

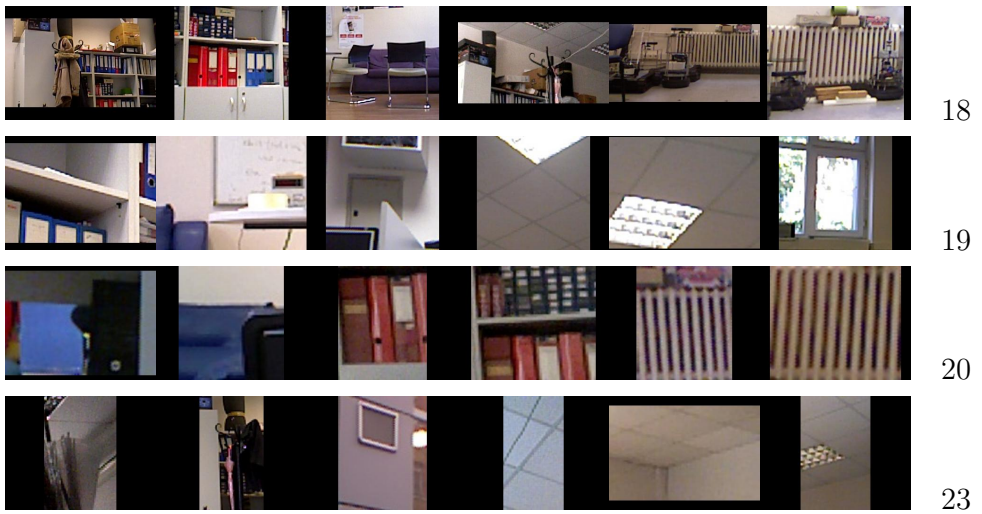
externally specified. It is observed that our approach has slightly better purity.

Table 3.4. New object categories in WOCM and their M_1 values using bubble descriptors. The number of categories N_h^* is either self-determined (U) or externally given (S).

$\mathcal{N}_w$	N_{h^*}	M_1	
		Ward's Method	K-means
3000	58	(U) 0.2133	(S) 0.2091
5000	68	(U) 0.2021	(S) 0.2002
3000	38	(S) 0.1863	(S) 0.1890
5000	38	(S) 0.1679	(S) 0.1680

The categorization performance with the evolving LOCM is shown in Table 3.6. Categorization precision is around 93% with recall around 50%. The robot ends up learning 105 object categories (terminal nodes) with average number of object candidates $\bar{\mu} = 65.89$ per category. The resulting structure of LOCM is shown in Fig. A.1(a). As expected, the categories of WOCM have low purities. For each category, the most common three objects that are represented by each learned category as determined through manual inspection are given in Table A.1-Table A.4. It is observed that some objects appear as top contender in several learned categories. The representations of true object categories with respect to the learned object categories are given in Table A.5. Some object categories (bookshelves) are learned in multiple categories. For example, there are 18 learned categories in which bookshelves are represented topmost. Among these, in 4 categories respectively, the representations of walls and chairs come in second. Closer inspection indicates that this tends to occur with larger objects. This may be attributed to the fact that they tend to get segmented into smaller parts or they exist in the background of discovered object candidates with different lighting, positions and surrounding objects. Some object categories such as person, phone and poster are not learned at all. These are possibly learned based on the neighboring objects. For example, posters may be learned as walls. There are also some objects which can only get in the second or third most. The main reason is the robot can confuse

(a) Highest M_1



(b) Lowest M_1

Figure 3.3. Sample new object categories in WOCM using bubble descriptors. Their learned categories in LOCM are as given.

them with objects such having similar appearances. For examples, switch boxes seem to be confused with monitors. Object candidates in sample categories are as shown in Figure 3.5.1. It is observed that brightness and texture are significantly important for the bubble descriptors. For example, 55th category seems to represent objects without texture such as walls, buildings and doors. On the other hand, category 40 has dark and highly textured objects such as bookshelves, hangers and umbrellas. Some of the categories are observed to represent more than three object classes, even though their number of samples may be comparably lower. For example, in Figure 3.5.1, while the learned category 75 contains some 'floor' samples, but the three top true categories

correspond to monitors, tables and ceiling regions.

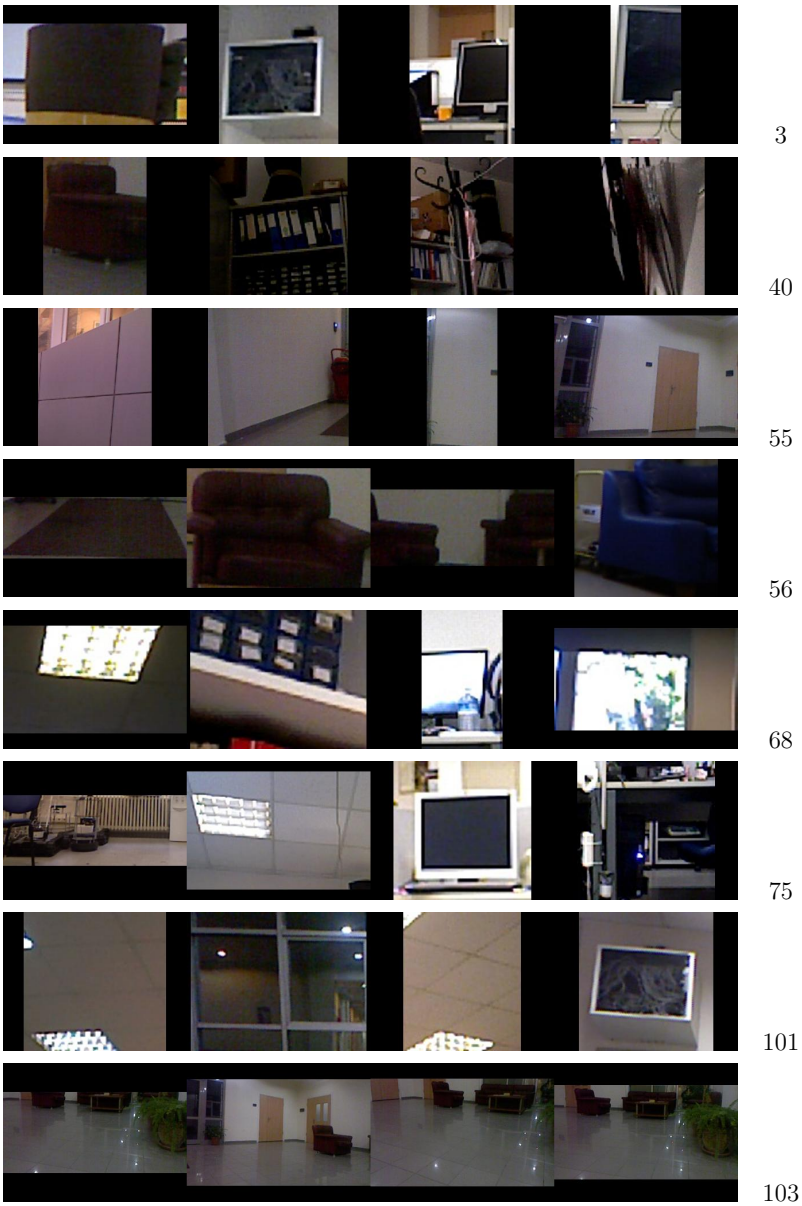


Figure 3.4. Sample object categories in LOCM using bubble descriptors

3.5.2. Comparative Results: Alexnet Descriptors

Alternatively, the object candidates are encoded by descriptors $o_c \in R^d$ with $d = 4096$ based on the f6 layer of the Alexnet. Thus, the features of this descriptor have been determined using supervised learning. The results are presented in Table 3.4 and Table 3.5. This suggests that the features encoded by the bubble descriptors need to be improved. The constructed LOCM is given in Figure A.2(a). The robot has

learned 133 object categories (terminal nodes) with an average number $\bar{\mu} = 33.9$ of object candidates per category. Thus, in comparison to the LOCM constructed with the bubble descriptors, more object categories are determined. Categorization precision is around 74% with recall around 49% as given in Table 3.6. These rates are lower which suggests that there are more unrecognized objects. On the other hand, the purity of the resulting object categories with respect to the 38 objects are better as shown in in Figure A.2(b) for alexnet descriptors. Some of the learned object categories have very high purity as seen in Table 3.5(a). However, there are also categories with low purities with samples as shown in Table 3.3(b).

Table 3.5. New object categories in WOCM using Alexnet descriptors. The number of categories N_h^* is either self-determined (U) or externally given (S).

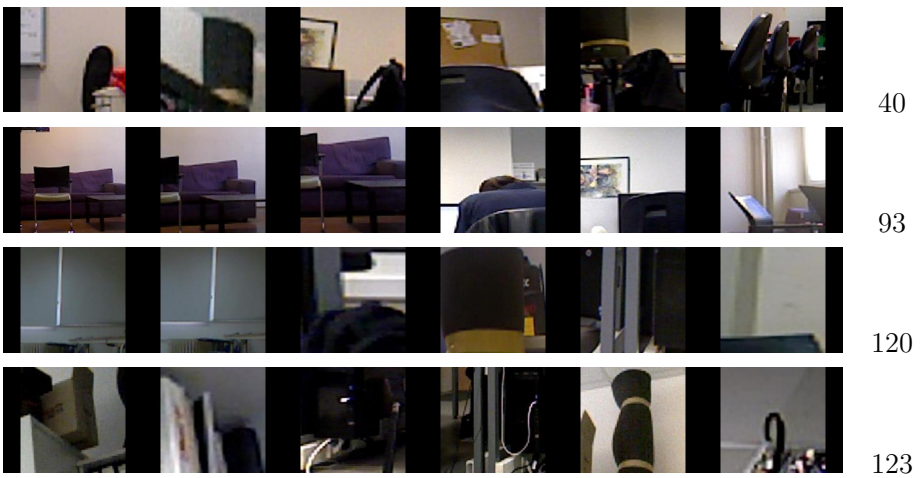
$\mathcal{N}_w$	N_{h^*}	M_1	
		Ward's Method	K-means
3000	143	(U) 0.5378	(S) 0.5294
5000	240	(U) 0.5199	(S) 0.5046
3000	38	(S) 0.4626	(S) 0.4383
5000	38	(S) 0.4427	(S) 0.4370

Table 3.6. Comparative categorization performance with the evolving LOCM.

Descriptor	N_o	$\bar{\mu}$	M_1	Precision	Recall	F1-Score
Bubble	105	65.89	0.1660	0.9307	0.5052	0.6549
Alexnet	133	33.9	0.4002	0.7425	0.4988	0.5967

In summary, the proposed model enables a robot to learn object categories on its own completely on its own. As such, it is usable in open-ended domains where learning has to be continual. Thus, the model constitutes a step towards having robots that can relate their surroundings to past experience. The two memories WOCM and LOCM have key roles in new category detection and new category learning. As expected, the descriptors used in encoding the incoming object candidates affect category learning and categorization performance. The features used in the bubble descriptors need to be improved as Alexnet descriptors yield categories with higher purities. However, as

(a) Highest M_1



(b) Lowest M_1

Figure 3.5. Sample object candidates from selected object categories in WOCM using Alexnet descriptors. They are added to the LOCM with the category numbers as given.

Alexnet descriptors are learned in a supervised manner, they are not appropriate for our system which is completely unsupervised. Rather, the features they encode may be used in selecting the filters that are encoded by the bubble descriptors.

4. OBJECT MAPPING

The construction of scene maps has been receiving significant attention. These are also referred to as objects based semantic maps. The resulting maps can be used for many different purposes such as object manipulation, searching and scene verification. In this chapter, the problem that is considered is to construct an object map and enable the robot to validate the surroundings of the robot to past experiences. The outline of the chapter is as follows: First, related literature is reviewed briefly in Section 4.1. In following general approach for construction of object map and validation is presented in Section 4.2. In Section 4.3 proposed method is detailed. In the last Section 4.4 experimental results are discussed.

4.1. Related Literature

The proposed methods vary along with a variety of considerations as presented in Table 4.1. The first consideration is related to the level of labeling namely whether it is done at the pixels or object candidates level. In the former, pixels are assigned categories and object instances are simply determined through the connected components analysis of the resulting labels [68–70]. In the latter, first, object candidates - namely blobs or segments that encode similar pixels in the incoming visual data - are generated [1, 2]. The robot then runs the categorization and or recognition algorithm on the generated object candidates one-by-one.

The next is regarding whether the temporal coherence of the incoming video data is taken advantage of or not. Most of the proposed approaches consider single shots of the scene and exploit each respective image by generating hundreds of object candidates to ensure complete scene coverage. The trade-off between computational tractability and high discovery quality is addressed by decreasing the number object candidates by using various 'objectness' measures [10, 11] or saliency-based approaches [6, 15] - motivated by related work in vision science [7]. Alternatively, the generation of object candidates has been considered on the level of the entire video sequence and

thus taking the temporal coherence of the video input into account. The proposed methods have primarily considered tracking individual region candidates over time with a focus on determining a small, yet consolidated number of object candidates [17, 18, 71, 72]. Alternatively, the saliency definition has been expanded to include a temporal dimension [12, 73–75]. However, in all these work, the video input is assumed to be externally generated and thus it is not clear how the robot actually generates the incoming visual data by itself. The generation of video input has been considered in visual exploration and active vision related work [20, 21, 23, 76] however as their focus is either on spatial exploration or exploration targeted to a single object. Interestingly, simply looking around through only moving the camera or head has not been considered in this context. Yet, since with a limited field of view camera, a single shot will not contain all the object candidates, looking around in space with a pan-tilt camera is the simplest means of visual exploration.

Table 4.1. Comparison of related work

Reference	Sensory Input	Processing Level	Temporal Nature	Looking Around	Objects' Learning	Coordinate Space
[68]	Stereo	Pixels	✗	✗	Supervised	Euclidean
[69]	Stereo	Pixels	✗	✗	Supervised	Euclidean
[70]	RGB + Laser	Pixels	✗	✗	Supervised	-
[77]	RGB-D	Segments	✗	✗	Supervised	Euclidean
Proposed	RGB	Segments	✔	✔	Unsupervised	Spherical

The third consideration is related to how categorization is done. Object candidates are encoded using a variety of representations ranging from using 'handcrafted' descriptors to those obtained from clustering image patches [33, 34] or from convolutional neural networks (CNNs) [35–37]. In most of the approaches, categorization is seen as a supervised problem. Most approaches like deep convolutional neural network methods are supervised methods and trained by labeled object candidates [38–41] . Unfortunately, there is a large number of objects with varying appearances [31].

4.2. Object Mapping: General Approach

The object map is constructed based on the object candidates that have been generated through scene exploration and recognized using the long-term object candidates

memory. The proposed approach differs from previous work in three aspects.

- The objects that are used to build scene graph are attained by taking advantage of the temporal coherence in the video for all possible object candidates as explained in previous Chapter 2 while exploring the scene autonomously.
- Additionally, unlike the supervised methods, our unsupervised evolving object recognition method is used for labeling each objects which is explained in Chapter 3.
- Finally our approach differs in coordinate space that is used in defining the object maps. The majority of mapping methods use various sensors that can provide depth information and the maps are mostly defined in some local Euclidean coordinate. In our case only a single RGB camera is used so depth of the objects is not available and mapping is performed in pan-tilt coordinate.

A validation method is introduced for comparing the object map with those previously constructed as retained in the robot's memory. This helps the robot to relate its surroundings to past experience.

4.3. Maps: Generation & Matching

The object map is defined by a graph of $\mathcal{G} = (\mathcal{N}, E)$ where nodes $\mathcal{N}$ correspond to the categorized objects and edges E encode their spatial relations - considering a complete graph. Each object is categorized as described in Chapter 3. The objects that cannot be categorized are not included in the maps. Their spatial relations are expressed in the pan-tilt space $\mathcal{F}$ that is used for object detection as described in Chapter 2. Since each object is associated with a pan $\mu(C)_1 \in S^1$ and tilt $\mu(C)_2 \in S^1$ angles, the spatial relations simply encode the relative pan and tilt angles of any pair of objects in the map.

The matching of two different object map is done by comparing the spatial relations of identically categorized objects in the two maps. As the spatial relations are represented in a robocentric coordinate system, relative angles between the objects

change depending on the distance of the robot to the objects around it. Since there is no knowledge of the robot's pose, the matching cannot be made through comparing the spatial relations directly. As such, a method that is based on comparing their relative quadrant positions is developed. This is because quadrant properties are preserved even if the robot's viewing position changes. Let $\mathcal{G}' = (\mathcal{N}', E')$ be graph to be matched. Let $o : \mathcal{N} \rightarrow \mathcal{O}$ the map that defines the object category of a given object in the map and $\beta_i : \mathcal{N} \rightarrow \mathcal{N}$ be the map such that for any $N \in \mathcal{N}$, $\beta_i(N)$ represents the objects in quadrant i with respect to object N. A matching score $\Upsilon(\mathcal{G}, \mathcal{G}')$ is defined as in Equation 4.1.

$$\Upsilon(\mathcal{G}, \mathcal{G}') = \frac{1}{|\mathcal{N}|} \sum_{N \in \mathcal{N}} \sum_{i=1}^{4} \frac{|\beta_i(N) \cap \beta_i(N')|}{min(|\mathcal{N}|, |\mathcal{N}'|)} \text{ where } o(N) = o(N') \tag{4.1}$$

4.4. Experimental Results

The proposed approach has been evaluated in three places through the manual inspection of the resulting object map as well as measuring their quality. Quality measurement is based on how well they match those of previous visits of the same scene or other scenes. Object detection and scene exploration is done as described in Chapter 2. The object candidates are internally encoded using bubble descriptors. Object categories are given as explained in Chapter 3.

For this, we consider three different places. These are chosen as a cluttered laboratory, uncluttered hall and outdoor. The robot visits the first two twice at different times. The object maps are generated at different locations and headings. For each visit, the number of generated object candidates and the number of categorized objects are listed in Table 4.2.

It is observed that outdoors place has the lowest rate. This is attributed that most of the experiments are done indoors so that the robot has more knowledge of indoors objects. Initially, the object maps are examined by comparing the most common object

classes in the assigned category. The most common three object classes in each category are as given in Table A.1-Table A.4.

Table 4.2. Generated and categorized object candidates.

	Visit #	# Generated Object Candidates	# Categorized Objects
Place 1	1	24	20
	2	28	25
Place 2	1	26	23
	2	34	29
Place 3	1	30	22

In the first visit of place 1, the robot covers a region of size $[-10.2°, 79.5°] \times [-20.1°, 15.2°]$ after 16 movements. The covered scene is shown in Figure A.3(a). This is a stitched image for visualization purposes. The resulting object map is as shown in Figure A.3(b). Through manual inspection, it is observed that 12 of the 20 categorized objects are correctly categorized. These correspond to categories corresponding to monitors, table parts, walls, chairs and computers. The remaining 8 are assigned to categories where the true category is not among the top three most observed. In particular, the following assignments are made: switch box - category 40, chair - category 6, two partial views of the table - category 26 and 26, floor parts - category 48 and 75 and table legs - category 101 and 101. For example, category 40 contains objects like a couch, ceiling and bookshelf. As such, it confuses a switch box with these. Similarly, it wrongly categorizes a chair to be in the same category as bookshelf, wall and curtain. In the second visit of place 1, its coverage is about $[-9.8°, 85.9°] \times [-21.8°, 5.6°]$ after 18 movements. The scene that is covered is shown in Figure A.4(a) with the constructed object map as shown in Figure A.4(b). Interestingly, categorization performance is worse here compared to the first visit. Only 8 objects are categorized correctly. These include objects from categories monitors, a wall, a table, a chair, a lower part of a table. The remaining are not correctly categorized: wall - category 66 , table legs - category 53 and 101, a jaguar robot - category 55, a box - category 5, a chair on the right - category 75, a computer case - category 5, two partial views of the table - categories 9 and 37, three parts of the chair - categories 66, 66 and 68, five parts of the floor - categories 5, 44, 44, 68, 75. Interestingly, some of the objects are observed to be

wrongly categorized in a coherent manner across different visits. For example, in both of the visits, floor regions are assigned to category 75 - which is primarily a category consists of monitors.

In the first visit of place 2, the robot covers $[-39.3°, 23.3°] \times [-15.2°, 21.6°]$ field of view with 15 movements. The covered scene is shown in Figure A.5(a). The object map consists of 23 objects as shown in Figure A.5(b). Through manual inspection, 15 objects are observed to be correctly categorized. These correspond to wall, floor, ceiling and couch class objects. The remaining are wrongly categorized. These are: two doors on the right side - categories 67 and 101 , four parts of floor - categories 40, 56, 56 and 57 and two parts of the outer door - categories 51 and 92. For examples, doors are found to be in the category containing bookshelves, drawers and walls or wall, ceiling and windows. When the robot revisits this place, it covers a scene of size $[-21.8°, 82.7°] \times [-19.7°, 18.5°]$ after 19 movements. The covered scene is shown in Figure A.6(a). The object map new contains 29 objects as seen in Figure A.6(b). Here, 14 objects are observed to be categorized correctly. These correspond floor, door, window, couch and wall categories. The remaining 14 are wrongly categorized. These are a plant - category 87, a wall - category 50, a part of window - category 61, an outer door - category 103, two parts of a door - categories 68, an 101, nine parts of the floor - categories 42, 40, 50, 50, 50, 55, 61, 65, 103. For example, both wall and window seem to be confused with each other.

Finally, the robot visits an outdoor place. In this case, it looks around in 14 movements and covers a region of $[-18.7°, 19.1°] \times [-17.6°, 23.2°]$ as shown in Figure A.7(a). It is able to categorize 22 objects as shown in Figure A.7(b). Again, through manual inspection, only 8 objects are observed to be correctly categorized. The remaining 14 objects are wrongly categorized. These include sidewalk - category 59, a wall - category 45, two buildings - categories 55 and 56 , 2 sky regions - categories 26 and 51, four road regions - categories 40, 87, 59 and 87 and four car areas - categories 56, 59, 71 and 77. Interestingly, sidewalk, road regions and car areas are all related to category 59 - which is related to windows, walls and black panel.

We also evaluate how well the object map constructed in the same place match and how much they differ from object maps that are constructed in different places. Of course, the performance will be affected by the fact that these maps contain wrongly categorized objects. Naturally, the robot does not know this. First, we compare the two object maps from place 1 through manual inspection. 4 objects are determined to be present in both of the visits - categories 3, 68, 75 and 101. However, these are not necessarily correct categorizations as discussed earlier. In this case, categories 3 and 68 are observed to be correct in both of the visits. Interestingly, 75 and 101 refer to the floor and table in both visits even though they are both wrongly categorized. When the object maps from the place 2 are examined, both maps are observed to have categories 40, 55, 56 and 103. Some of these category assignments are not correct. For example, the object with category 103 corresponds to the outer door in visit 1, but the ground truth for this object in the other visit is different. In this case, categories 56 and 103 are observed to be correct in both of the visits. Interestingly, while objects corresponding to categories 40 and 55 are actually wrongly categorized, this is consistently done in both of the visits.

Table 4.3. Matching object maps.

		Place 1		Place 2		Place 3
	Scene	1	2	1	2	1
Place 1	1	1	0.1645	0	0.1092	0
	2	0.1645	1	0.1023	0.1533	0
Place 2	1	0	0.1023	1	0.1146	0.0997
	2	0.1092	0.1533	0.1146	1	0.0588

There are also shared labels among the scenes from different places too. However, their position differs in the object graph. For example, objects with label 40 is exist in all three places but it is located considerably differ in graph positions. In scene 1 of the place 1, it is located in top left and in scene 1 of place 2, it is located at lower-center. Similarly, in the other scene of place 2 it is located in most bottom of the center. In the third place it is located at the bottom left side of the graph. In the Table 4.3, the performance of the validation formula is shown. Even though scores are close to each

other, the only miss match is between scene 2 of the place 2 and the scene 2 of the place 1. The main reason is shared labels have a more similar positioning. Additionally, the scenes from the place 1 match to each other, additionally scene 1 of the place 2 matches to the scene 2 of the place 2. The validation score of the outdoor scene is below 0.1.

5. CONCLUSION

This thesis is concerned with constructing object-based scene maps and validation using a RGB pan-tilt camera placed on a robot. This is achieved by solving a sequence of distinct, yet related problems - namely the generation of object candidates, learning and recognizing object categories and finally constructing the object map and using it for validation.

The first contribution of this thesis is related to the generation of object candidates. The novelty of the proposed approach is that it simultaneously enables the robot to look around and to take advantage of the temporal coherence of the incoming video data in generating the object candidates. The robot's camera movements are governed by the set of object candidates that have been generated, but not directly looked at. In parallel, the robot discovers the object candidates from the incoming video by determining the spatio-temporally coherent segments. The generated object candidates turn out to be combined across the incoming visual stream as demonstrated in the experimental results.

The second contribution is related to the learning and recognition of object categories. For this, a novel learning approach is proposed. This approach introduces two novelties: First, the number of levels and the structure of the hierarchy associated with the long-term object categories memory(LOCM) are completely free and evolve as necessary depending on the robot's visual experiences. Second, the clusters of object candidates that are determined without any supervision in the working object candidates' memory are then sent as the basis for the open-ended learning of object categories in the LOCM. Our experimental results indicate that the robot is able to determine new object categories with an acceptable level of impurity and then to evolve its LOCM without any supervision.

Lastly, the construction of object maps are constructed. The map is defined by a complete graph of the categorized objects in the scene. Our map differs from

the most since the edges of this map are defined in robocentric spherical coordinate space. In order to be able to match object maps with each other, a matching method is introduced.

All the proposed approaches are evaluated on data obtained with a mobile robot endowed with a pan-tilt RGB camera. Experimental results demonstrate that the robot is able to generate object candidates as necessary. It is also able to learn different categories on their own and recognize them as necessary. Finally, the resulting objects maps enable the semantic analysis of the scene. For future, this work can be extended by considering the body movement of the robot and using additional sensory such as depth.

REFERENCES

1. Tuytelaars, T., C. H. Lampert, M. B. Blaschko and W. Buntine, "Unsupervised object discovery: A comparison", *International Journal of Computer Vision*, Vol. 88, No. 2, pp. 284–302, 2010.

2. Hosang, J., R. Benenson and B. Schiele, "How good are detection proposals, really?", *British Machine Vision Conference*, 2014.

3. Marfil, R., A. J. Palomino and A. Bandera, "Combining segmentation and attention: a new foveal attention model", *Frontiers in computational neuroscience*, Vol. 8, 2014.

4. Driver, J., G. Davis, C. Russell, M. Turatto and E. Freeman, "Segmentation, attention and phenomenal visual objects", *Cognition*, Vol. 80, No. 1-2, pp. 61–95, 2001.

5. Martín García, G., T. Werner and S. Frintrop, "Attentional Scene-Exploration and Object Discovery in Image and RGB-D Data", *KI - Künstliche Intelligenz*, Vol. 29, No. 1, pp. 75–81, 2015.

6. Walther, D. and C. Koch, "Modeling attention to salient proto-objects", *Neural Networks*, Vol. 19, No. 9, pp. 1395 – 1407, 2006.

7. Scholl, B. J., "Objects and attention: the state of the art", *Cognition*, Vol. 80, No. 1–2, pp. 1 – 46, 2001.

8. Hwang, A. D., H.-C. Wang and M. Pomplun, "Semantic guidance of eye movements in real-world scenes", *Vision Research*, Vol. 51, No. 10, pp. 1192 – 1205, 2011.

9. Pajak, M. and A. Nuthmann, "Object-based saccadic selection during scene perception: Evidence from viewing position effects", *Journal of Vision*, Vol. 13, No. 5,

p. 2, 2013.

10. Alexe, B., T. Deselaers and V. Ferrari, "Measuring the objectness of image windows", *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, Vol. 34, No. 11, pp. 2189–2202, 2012.

11. Endres, I. and D. Hoiem, "Category-Independent Object Proposals with Diverse Ranking", *IEEE Transaction on Pattern Analysis and Machine Intelligence*, Vol. 36, No. 2, pp. 222–234, February 2014.

12. Mishra, A. K., Y. Aloimonos, L. F. Cheong and A. Kassim, "Active Visual Segmentation", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 34, No. 4, pp. 639–653, 2012.

13. Meger, D., P.-E. Forssén, K. Lai, S. Helmer, S. McCann, T. Southey, M. Baumann, J. J. Little and D. G. Lowe, "Curious george: An attentive semantic robot", *Robotics and Autonomous Systems*, Vol. 56, No. 6, pp. 503–511, 2008.

14. Orabona, F., G. Metta and G. Sandini, "A proto-object based visual attention model", *WAPCV*, pp. 198–205, 2007.

15. Marfil, R., A. J. Palomino and A. Bandera, "Combining segmentation and attention: a new foveal attention model", *Frontiers in Computational Neuroscience*, Vol. 8, p. 96, 2014.

16. Pillai, S. and J. Leonard, "Monocular SLAM Supported Object Recognition", *Robotics: Science and Systems*, Rome, Italy, July 2015.

17. Ma, L. and G. Sibley, "Unsupervised Dense Object Discovery, Detection, Tracking and Reconstruction", D. Fleet, T. Pajdla, B. Schiele and T. Tuytelaars (Editors), *European Conferance on Computer Vision*, pp. 80–95, 2014.

18. Horbert, E., G. M. García, S. Frintrop and B. Leibe, "Sequence-level object can-

didates based on saliency for generic object recognition on mobile systems", *IEEE International Conferance on Robotics Automation*, pp. 127–134, 2015.

19. Ramos, G. R., K. Kaspar, S. König, S. Nordholt and P. König, "Exploration and Exploitation in Natural Viewing Behavior", *Scientific Reports*, Vol. 7, No. 2311, 2017.

20. Roy, S. D., S. Chaudhury and S. Banerjee, "Active recognition through next view planning: A survey", *Pattern Recognition*, Vol. 37, No. 3, pp. 429 – 446, 2004.

21. Atanasov, N., B. Sankaran, J. L. Ny, G. J. Pappas and K. Daniilidis, "Nonmyopic View Planning for Active Object Classification and Pose Estimation", *IEEE Transaction on Robotic*, Vol. 30, No. 5, pp. 1078–1090, 2014.

22. Eidenberger, R. and J. Scharinger, "Active perception and scene modeling by planning with probabilistic 6D object poses", *International Conference on Intelligent Robots and Systems*, pp. 1036–1043, 2010.

23. Sommerlade, E. and I. Reid, "Information-theoretic active scene exploration", *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–7, 2008.

24. Akdeniz, B. C. and H. I. Bozma, "Exploration and topological map building in unknown environments", *IEEE International Conferance on Robotics and Automation*, pp. 1079–1084, 2015.

25. Kim, A. and R. M. Eustice, "Active visual SLAM for robotic area coverage: Theory and experiment", *The International Journal of Robotics Research*, Vol. 34, No. 4-5, pp. 457–475, 2015.

26. Odabaşı, Ç., *Constructing Semantic Place Representation Via Object Discovery and Visual Exploration*, M.S. Thesis, Boğaziçi University, 2017.

27. Erkent, Ö. and H. I. Bozma, "Artificial potential functions based camera move-

ments and visual behaviors in attentive robots", *Autonomous Robots*, Vol. 32, No. 1, pp. 15–34, 2012.

28. Bindemann, M., "Scene and screen center bias early eye movements in scene viewing", *Vision Research*, Vol. 50, No. 23, pp. 2577 – 2587, 2010.

29. Felzenszwalb, P. F. and D. P. Huttenlocher, "Efficient graph-based image segmentation", *International Journal of Computer Vision*, Vol. 59, No. 2, pp. 167–181, 2004.

30. Yao, R., G. Lin, S. Xia, J. Zhao and Y. Zhou, "Video Object Segmentation and Tracking: A Survey", *Computing Research Repository*, Vol. abs/1904.09172, 2019.

31. Biederman, I., "Recognition-by-components: a theory of human image understanding.", *Psychological review*, Vol. 94, No. 2, p. 115, 1987.

32. Dean, T., M. A. Ruzon, M. Segal, J. Shlens, S. Vijayanarasimhan and J. Yagnik, "Fast, Accurate Detection of 100,000 Object Classes on a Single Machine", *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1814–1821, 2013.

33. Coates, A. and A. Y. Ng, "Learning feature representations with k-means", *Neural Networks: Tricks of the Trade*, pp. 561–580, Springer, 2012.

34. Oliveira, M., L. S. Lopes, G. H. Lim, S. H. Kasaei, A. D. Sappa and A. M. Tomé, "Concurrent learning of visual codebooks and object categories in open-ended domains", *Intelligent Robots and Systems*, pp. 2488–2495, 2015.

35. Ranzato, M., F. J. Huang, Y. Boureau and Y. LeCun, "Unsupervised Learning of Invariant Feature Hierarchies with Applications to Object Recognition", *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1–8, 2007.

36. Sivic, J., B. C. Russell, A. Zisserman, W. T. Freeman and A. A. Efros, "Unsupervised discovery of visual object class hierarchies", *Conference on Computer Vision*

and Pattern Recognition, pp. 1–8, June 2008.

37. Le, Q. V., "Building high-level features using large scale unsupervised learning", *IEEE International Conference On Acoustics, Speech and Signal Processing*, pp. 8595–8598, 2013.

38. Fidler, S. and A. Leonardis, "Towards Scalable Representations of Object Categories: Learning a Hierarchy of Parts", *Conference on Computer Vision and Pattern Recognition*, pp. 1–8, June 2007.

39. Yan, Z., V. Jagadeesh, D. DeCoste, W. Di and R. Piramuthu, "HD-CNN: Hierarchical Deep Convolutional Neural Network for Image Classification", *Computing Research Repository*, Vol. abs/1410.0736, 2014.

40. Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke and A. Rabinovich, "Going deeper with convolutions", *Conference on Computer Vision and Pattern Recognition*, pp. 1–9, 2015.

41. Simonyan, K. and A. Zisserman, "Very deep convolutional networks for large-scale image recognition", *arXiv:1409.1556*, 2014.

42. Eitel, A., J. T. Springenberg, L. Spinello, M. Riedmiller and W. Burgard, "Multimodal deep learning for robust RGB-D object recognition", *Conference on Intelligent Robots and Systems*, pp. 681–687, 2015.

43. He, K., X. Zhang, S. Ren and J. Sun, "Spatial pyramid pooling in deep convolutional networks for visual recognition", *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, Vol. 37, No. 9, pp. 1904–1916, 2015.

44. Huang, H., Y. Chuang and C. Chen, "Affinity aggregation for spectral clustering", *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 773–780, 2012.

45. Girshick, R., J. Donahue, T. Darrell and J. Malik, "Rich Feature Hierarchies for Ac-

curate Object Detection and Semantic Segmentation", *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 580–587, 2014.

46. Ren, S., K. He, R. Girshick and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 39, No. 6, pp. 1137–1149, June 2017.

47. Kinnunen, T., J. Kamarainen, L. Lensu and H. Kalviainen, "Unsupervised Visual Object Categorisation via Self-organisation", *International Conference on Pattern Recognition*, pp. 440–443, 2010.

48. Tian, F., B. Gao, Q. Cui, E. Chen and T.-Y. Liu, "Learning Deep Representations for Graph Clustering", *American Association For Applied Linguistics, Conference on Artificial Intelligence*, pp. 1293–1299, 2014.

49. Trigeorgis, G., K. Bousmalis, S. Zafeiriou and B. W. Schuller, "A Deep semi-NMF Model for Learning Hidden Representations", *International Conference on Machine Learning*, Vol. 32, pp. II–1692–II–1700, 2014.

50. Wang, Z., S. Chang, J. Zhou and T. S. Huang, "Learning A Task-Specific Deep Architecture For Clustering", *Computing Research Repository*, Vol. abs/1509.00151, 2015.

51. Xie, P. and E. P. Xing, "Integrating Image Clustering and Codebook Learning", *American Association For Applied Linguistics*, 2015.

52. Yang, J., D. Parikh and D. Batra, "Joint Unsupervised Learning of Deep Representations and Image Clusters", *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5147–5156, 2016.

53. Zweig, A. and D. Weinshall, "Exploiting Object Hierarchy: Combining Models from Different Category Levels", *ICCV*, pp. 1–8, Oct 2007.

54. Zhang, J., J. Zhang and S. Chen, "Discover Novel Visual Categories From Dynamic Hierarchies Using Multimodal Attributes", *IEEE Transactions on Industrial Informatics*, Vol. 9, No. 3, pp. 1688–1696, 2013.

55. Erkent, Ö. and H. I. Bozma, "Bubble space and place representation in topological maps", *The International Journal of Robotics Research*, Vol. 32, No. 6, pp. 672–689, 2013.

56. Krizhevsky, A., I. Sutskever and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks", F. Pereira, C. J. C. Burges, L. Bottou and K. Q. Weinberger (Editors), *Advances in Neural Information Processing Systems 25*, pp. 1097–1105, Curran Associates, Inc., 2012.

57. Ward, J., J. H., "Hierarchical Grouping to Optimize an Objective Function", *Journal of the American Statistical Association*, Vol. 58, p. 236–244, 1963.

58. Murtagh, F. and P. Legendre, "Ward's hierarchical agglomerative clustering method: which algorithms implement Ward's criterion?", *Journal of classification*, Vol. 31, No. 3, pp. 274–295, 2014.

59. Salvador, S. and P. Chan, "Determining the number of clusters/segments in hierarchical clustering/segmentation algorithms", *IEEE International Conference on Tools with Artificial Intelligence*, pp. 576–584, 2004.

60. Sibson, R., "SLINK: An optimally efficient algorithm for the single-link cluster method", *The Computer Journal*, Vol. 16, No. 1, pp. 30–34, 1973.

61. Takahashi, F. and S. Abe, "Decision-tree-based multiclass support vector machines", *International Conference on Neural Information Processing*, Vol. 3, pp. 1418–1422, Nov 2002.

62. Bennani, Y. and K. Benabdeslem, "Dendogram-based SVM for multi-class classification", *Journal of Computing and Information Technology*, Vol. 14, No. 4, pp.

283–289, 2006.

63. Bala, M. and R. Agrawal, "Optimal Decision Tree Based Multi-class Support Vector Machine", *Informatica (Ljubljana)*, Vol. 35, 01 2011.

64. Thiery, T., T. Lajnef, K. Jerbi, M. Arguin, M. Aubin and P. Jolicoeur, "Decoding the Locus of Covert Visuospatial Attention from EEG Signals", *Public Library of Science One*, 2016.

65. Schölkopf, B., R. Williamson, A. Smola, J. Shawe-Taylor and J. Platt, "Support Vector Method for Novelty Detection", *International Conference on Neural Information Processing Systems*, pp. 582–588, 1999.

66. Chauhan, A. and L. Seabra Lopes, "Using spoken words to guide open-ended category formation", *Cognitive Processing*, Vol. 12, No. 4, pp. 341–354, 2011.

67. Lancichinetti, A., S. Fortunato and J. Kertész, "Detecting the overlapping and hierarchical community structure in complex networks", *New Journal of Physics*, Vol. 11, No. 3, p. 033015, March 2009.

68. Ranganathan, A. and F. Dellaert, "Semantic modeling of places using objects", *Proceedings of the 2007 Robotics: Science and Systems Conference*, Vol. 3, pp. 27–30, 2007.

69. Vasudevan, S., S. Gächter, V. Nguyen and R. Siegwart, "Cognitive maps for mobile robots - an object based approach", *Robotics and Autonomous Systems*, Vol. 55, pp. 359–371, 2007.

70. Zender, H., O. Mozos, P. Jensfelt, G.-J. Kruijff and W. Burgard, "Conceptual Spatial Representations for Indoor Mobile Robots", *Robotics and Autonomous Systems*, Vol. 56, pp. 493–502, 06 2008.

71. Hampapur, A., L. Brown, J. Connell, A. Ekin, N. Haas, M. Lu, H. Merkl and

S. Pankanti, "Smart video surveillance: exploring the concept of multiscale spatiotemporal tracking", *IEEE Signal Processing Magazine*, Vol. 22, No. 2, pp. 38–51, 2005.

72. Demir, M. and H. I. Bozma, "Video Summarization via Segments Summary Graphs", *International Conference on Computer Vision: Video Summarization for Large-Scale Analytics Workshop*, pp. 19–25, 2015.

73. Marat, S., T. Ho-Phuoc, L. Granjon, N. Guyader, D. Pellerin and A. Guérin-Dugué, "Modelling Spatio-Temporal Saliency to Predict Gaze Direction for Short Videos", *International Journal of Computer Vision*, Vol. 82, 05 2009.

74. Seo, H. and P. Milanfar, "Static and Space-time Visual Saliency Detection by Self-Resemblance", *Journal of vision*, Vol. 9, pp. 15.1–27, 11 2009.

75. Orabona, F., G. Metta and G. Sandini, "A proto-object based visual attention model", *International Workshop on Attention in Cognitive Systems*, pp. 198–215, Springer, 2007.

76. Chen, S., Y. Li and N. Kwok, "Active Vision in Robotic Systems: A Survey of Recent Developments", *The International Journal of Robotics Research*, Vol. 30, pp. 1343–1377, 10 2011.

77. Anand, A., H. Koppula, T. Joachims and A. Saxena, "Contextually Guided Semantic Labeling and Search for 3D Point Clouds", *The International Journal of Robotics Research*, Vol. 32, 11 2011.

78. Quigley, M., B. Gerkey, K. Conley, J. Faust, T. Foote, J. Leibs, E. Berger, R. Wheeler and A. Ng, "ROS: an open-source Robot Operating System", *International Conference on Robotics and Automation Workshop on Open Source Robotics*, Kobe, Japan, May 2009.

79. Jia, Y., E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadar-

rama and T. Darrell, "Caffe: Convolutional Architecture for Fast Feature Embedding", *arXiv preprint arXiv:1408.5093*, 2014.

80. Bradski, G., "The OpenCV Library", *Dr. Dobb's Journal of Software Tools*, 2000.

APPENDIX A: TABLES AND FIGURES

Table A.1. Mostly represented objects in learned object categories 1-22

Learned category	Top three true categories		
	1	2	3
1	Bookshelf	Chair	Drawer
2	Computer Case	Building	Chair
3	Window	Table	Monitor
4	Wall	Ceiling	Black Panel
5	Central Heating	Table	Hanger
6	Bookshelf	Wall	Curtain
7	Ceiling	Wall	Light Fixture
8	Wall	Central Heating	Window
9	Bookshelf	Curtain	Wardrobe
10	Bookshelf	Ceiling	Window
11	Wall	Drawer	Cylinder
12	Drawer	Wall	White Board
13	Bookshelf	Computer Case	Table
14	Couch	Black Panel	Wall
15	Cylinder	Ceiling	Couch
16	Ceiling	Light Fixture	Wall
17	Ceiling	Light Fixture	Wall
18	Bookshelf	Hanger	Table
19	Light Fixture	Ceiling	Bookshelf
20	Bookshelf	Ceiling	Computer Case
21	Wall	Ceiling	Bookshelf
22	Ceiling	Table	Window

Table A.2. Mostly represented objects in learned object categories 23-48

Learned category	Top three representations		
	1	2	3
23	Ceiling	Building	Floor
24	Building	Wall	White Board
25	Light Fixture	Ceiling	Wall
26	Ceiling	Wall	Bookshelf
27	Drawer	Box	Couch
28	Table	Bookshelf	Umbrella
29	Bookshelf	Chair	Window
30	Chair	Computer Case	Bookshelf
31	Curtain	Wall	Wardrobe
32	Window	Wall	White Board
33	Wall	Curtain	Ceiling
34	Ceiling	Wall	Floor
35	Chair	Computer Case	Monitor
36	Bookshelf	Table	Chair
37	Black Panel	Bookshelf	Building
38	Bookshelf	Air-conditioner	Cylinder
39	Table	Monitor	Bookshelf
40	Couch	Ceiling	Bookshelf
41	Bookshelf	Curtain	Drawer
42	Ceiling	Couch	Wall
43	Wall	Black Panel	Floor
44	Bookshelf	Wall	Monitor
45	Central Heating	Floor	Road
46	Couch	Car	Box
47	Bookshelf	Chair	Computer Case
48	Door	Window	Central Heating

Table A.3. Mostly represented objects in learned object categories 49-76

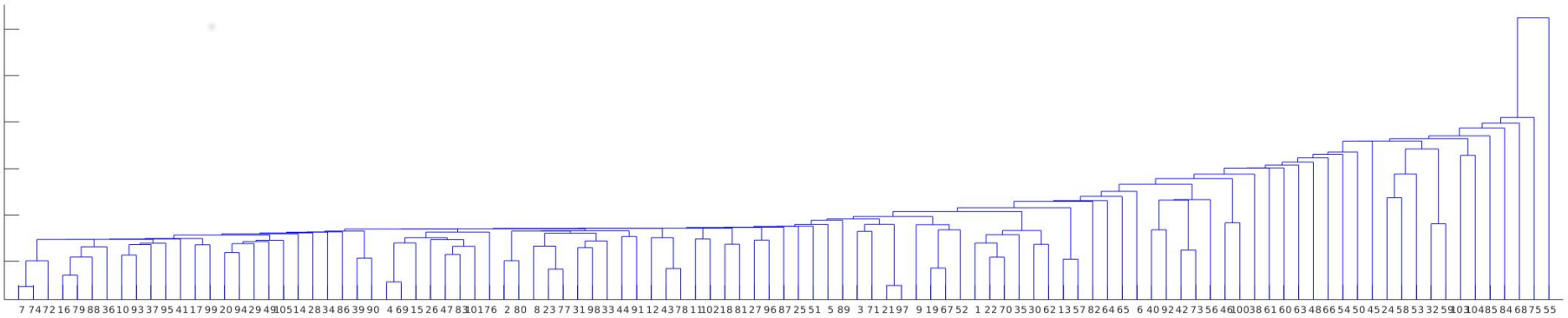
Learned category	Top three representations		
	1	2	3
49	Ceiling	Wall	Curtain
50	Window	Black Panel	-
51	Couch	Car	Sidewalk
52	Road	Floor	Ceiling
53	Window	Building	Wall
54	Wall	Door	Floor
55	Wall	Window	Door
56	Couch	Floor	Fuse Box
57	Couch	Box	Door
58	Wall	Building	Window
59	Window	Wall	Black Panel
60	Wall	Curtain	Drawer
61	Ceiling	Box	Cylinder
62	Computer Case	Chair	Table
63	Curtain	Dressing Screen Frame	Cylinder
64	Curtain	Wardrobe	Box
65	Dressing Screen Frame	Cylinder	-
66	Floor	Window	Curtain
67	Drawer	Bookshelf	Wall
68	Window	Table	Monitor
69	Bookshelf	Window	Wall
70	Bookshelf	Table	Chair
71	Bookshelf	Chair	Cylinder
72	Ceiling	Wall	Light Fixture
73	Ceiling	Couch	Window
74	Ceiling	Light Fixture	Wall
75	Monitor	Table	Ceiling
76	Chair	Computer Case	Table

Table A.4. Mostly represented objects in learned object categories 77-105

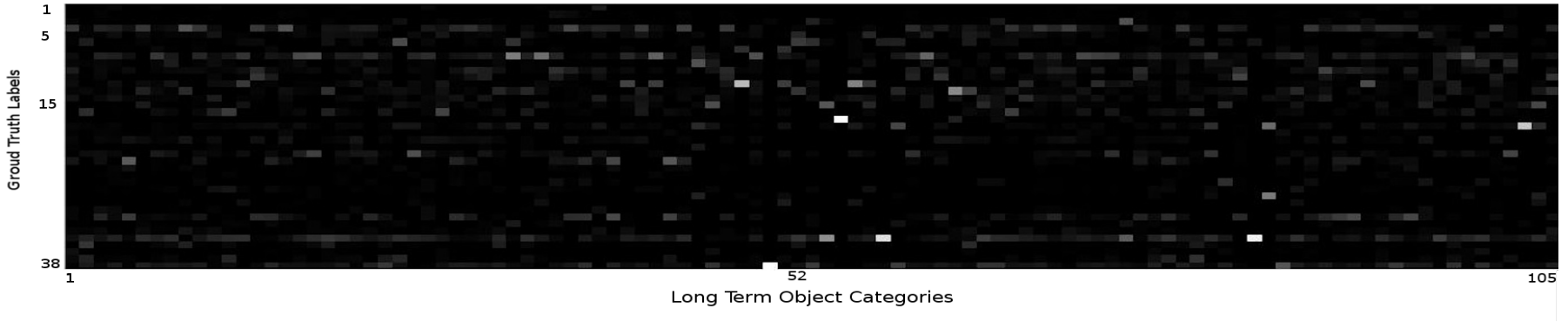
Learned category	Top three representations		
	1	2	3
77	Ceiling	Wall	Window
78	Wall	White Board	Monitor
79	Bookshelf	Window	Wall
80	White Board	Drawer	Building
81	Light Fixture	Ceiling	Floor
82	Plastic Bag	Door	Bookshelf
83	Computer Case	Chair	Table
84	Wall	Building	-
85	Building	Curtain	Wall
86	Window	Ceiling	Wall
87	Wall	Curtain	Ceiling
88	Bookshelf	Table	Bookshelf
89	Chair	Table	Computer Case
90	Table	Bookshelf	Monitor
91	Couch	Ceiling	Wall
92	Table	Chair	Monitor
93	Bookshelf	Wall	Window
94	Window	Bookshelf	Table
95	Table	Bookshelf	Chair
96	Box	Door	Window
97	Wall	Ceiling	Drawer
98	Window	Ceiling	Wall
99	Ceiling	Wall	Car
100	Wall	Window	Central Heating
101	Wall	Ceiling	Window
102	Light Fixture	Central Heating	Ceiling
103	Floor	Door	Tree
104	Door	Floor	Wall
105	Computer Case	Curtain	Bookshelf

Table A.5. True categories and the corresponding learned categories

True category	Corresponding learned categories		
	1	2	3
Air Conditioner	-	38	-
Backpack	-	-	-
Black Pane	37	14,43,50	4,59
Bookshelf	1,6,9,10,13,18,20,29,36,38,41,44,47,69,70,71,79,88,93	28,37,67,90,94,95	19,26,30,39,40,82,105
Box	96	27,57,61	46,64
Building	24,85	2,23,53,58,84	37,80
Car	-	46,51	99
Ceiling	7,16,17,22,23,26,34,42,49,61,72,73,74,77,99	4,,10,15,19,20,21,25,40,81,86,91,97,98,101	33,52,75,87,102
Central Heating	5,45	8,102	48,100
Chair	30,35,76,89	1,29,47,62,71,83,92	2,36,70,95
Computer case	2,62,83,105	13,30,35,76	20,47,89
Couch	14,40,46,51,56,57,91	42,73	15,27
Curtain	31,63,64	9,33,41,49,60,85,87,105	6,66
Cylinder	15	65,71	11,38,61,63
Door	48	54,82,96,103	55,57
Drawer	12,27,67	11,80	1,41,60,97
Dressing screen frame	65	63	-
Floor	38,66,103	45,52,56,104	23,34,43,54,81
Fuse box	-	-	56
Hanger	-	18	5,88
Jaguar robot	-	-	-
Light fixture	19,25,81,102	16,17,74	7,72
Monitor	75	39	3,35,44,68,78,90,92
Person	-	-	-
Phone	-	-	-
Plastic bag	82	-	-
Poster	-	-	-
Road	52	-	45
Sidewalk	-	-	51
Switch box	-	-	-
Table	28,39,90,92,95	3,5,22,36,68,70,75,88,89	13,18,62,76,83,94
Tree	-	-	103
Umbrella	-	-	28
Wall	4,8,11,21,33,43,54,55,58,60,78,84,87,97,100,101	6,7,12,24,26,31,32,34,44,49,59,72,77,93,99	14,16,17,25,42,53,67,69,74,79,85,86,91,98,104
Wardrobe	-	64	9,31
Water fountain	-	-	-
White board	80	78	12,24,32
Window	3,32,50,53,59,68,86,94,98	48,55,66,69,79,100	8,10,22,29,58,73,77,93,96,101

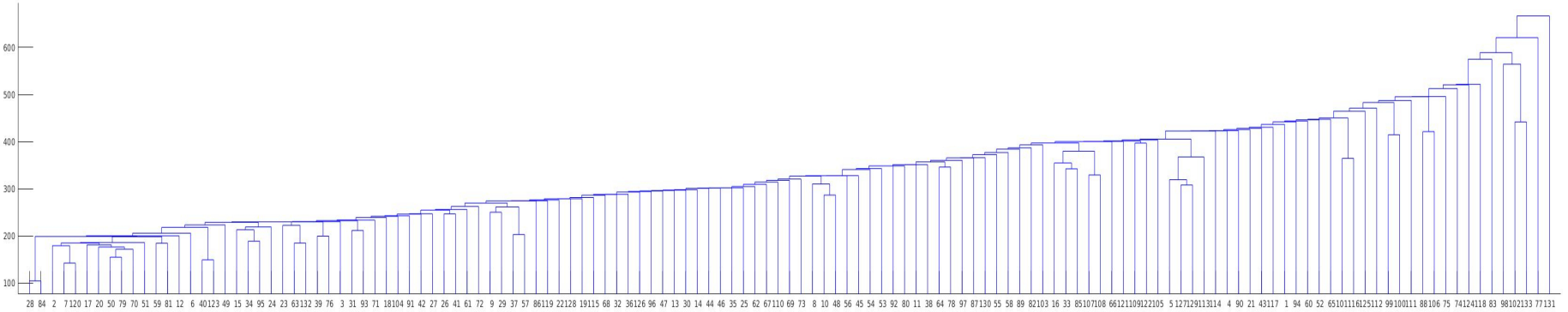


(a) LOCM hierarchy

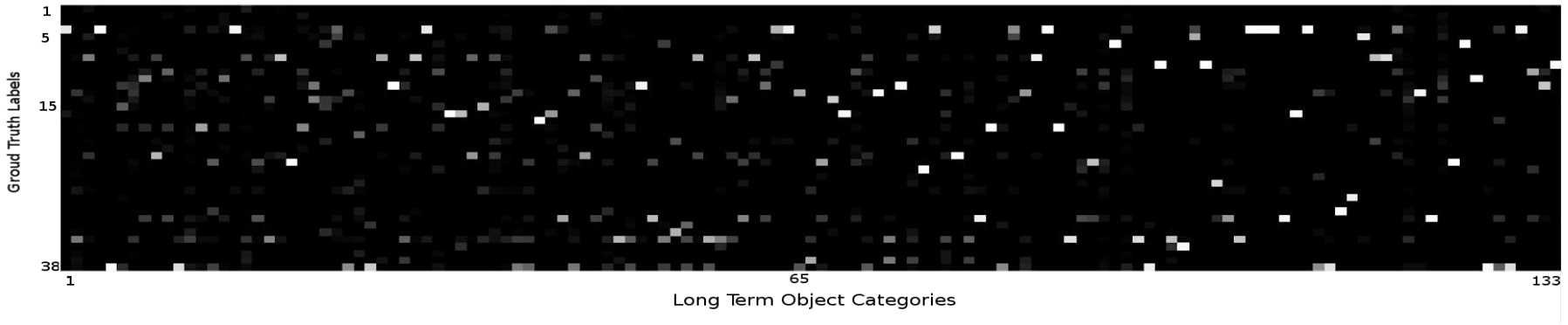


(b) Purity of learned object categories wrt 38 objects. White indicates 100% while black indicates 0%.

Figure A.1. LOCM using bubble descriptors

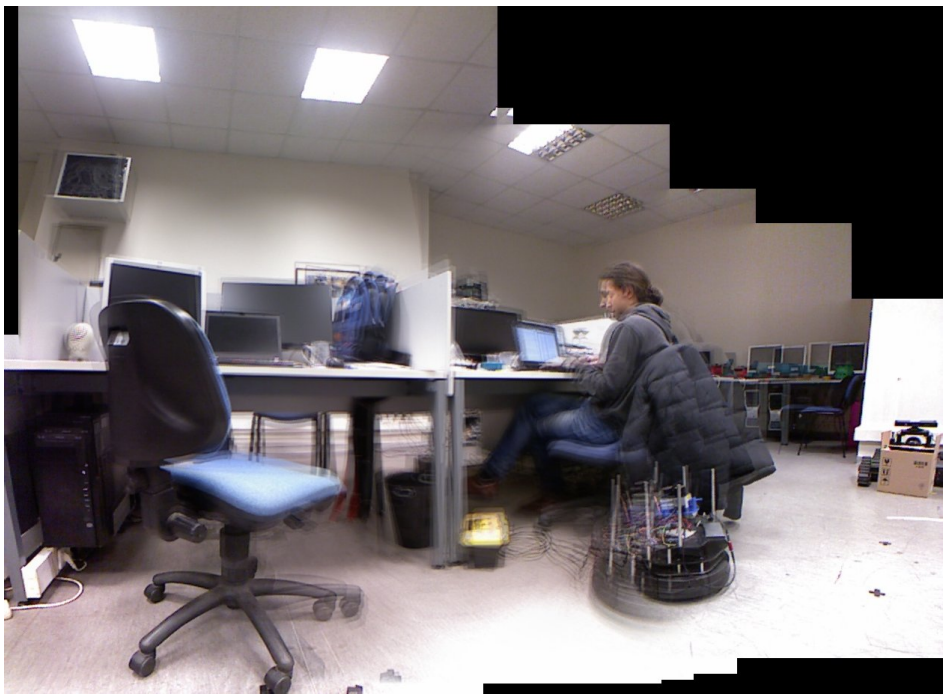


(a) LOCM hierarchy

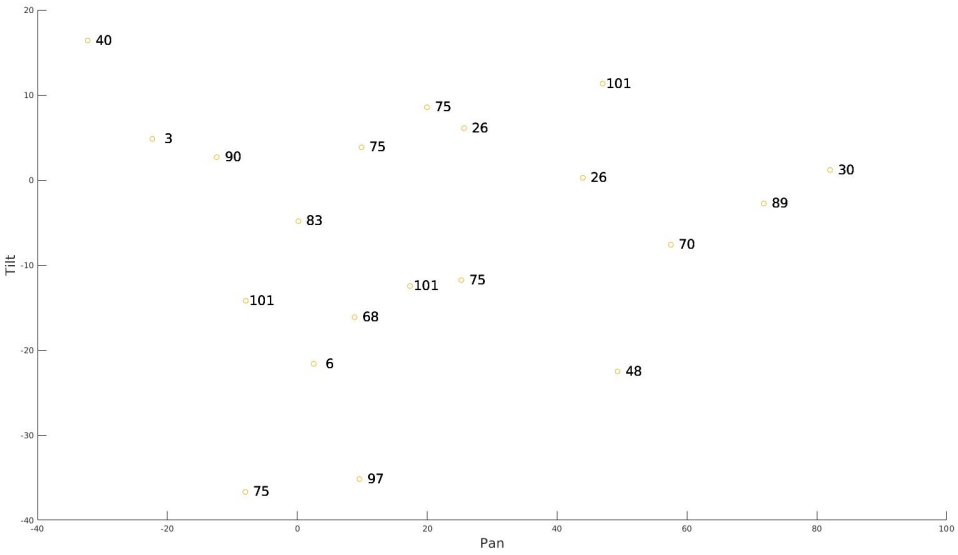


(b) Purity of learned object categories wrt true 38 categories. White indicates 100% while black indicates 0%.

Figure A.2. LOCM using Alexnet descriptors

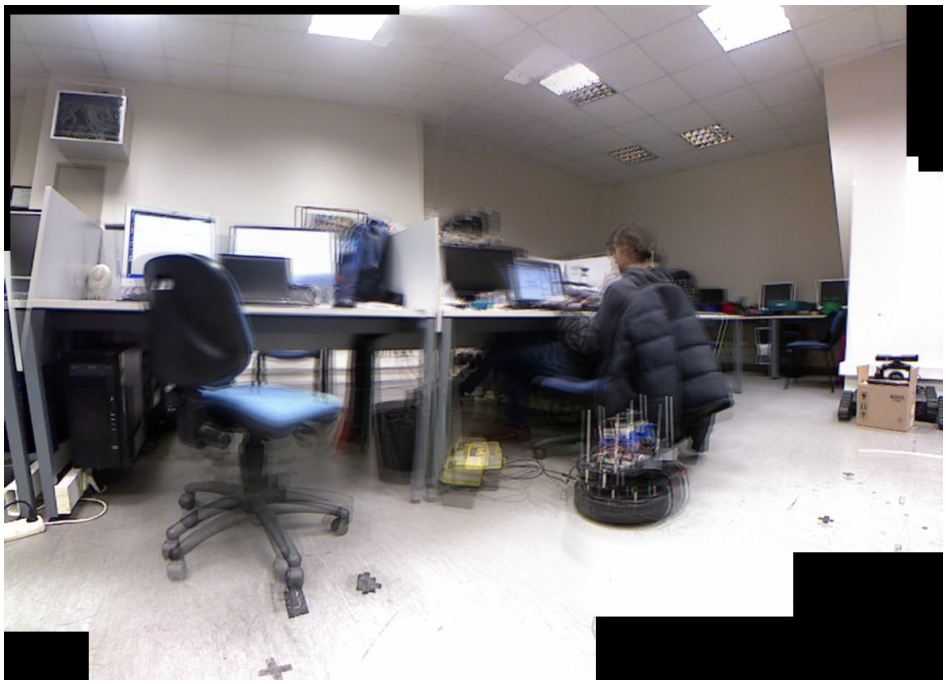


(a) Place 1 - Visit 1

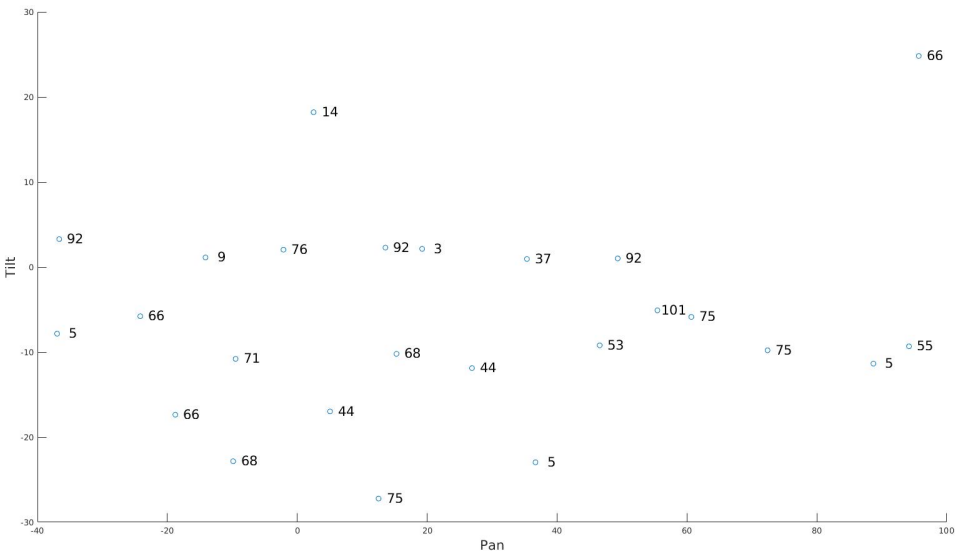


(b) Place 1 - Visit 1 Map

Figure A.3. Visual and the object map of place 1 - visit 1



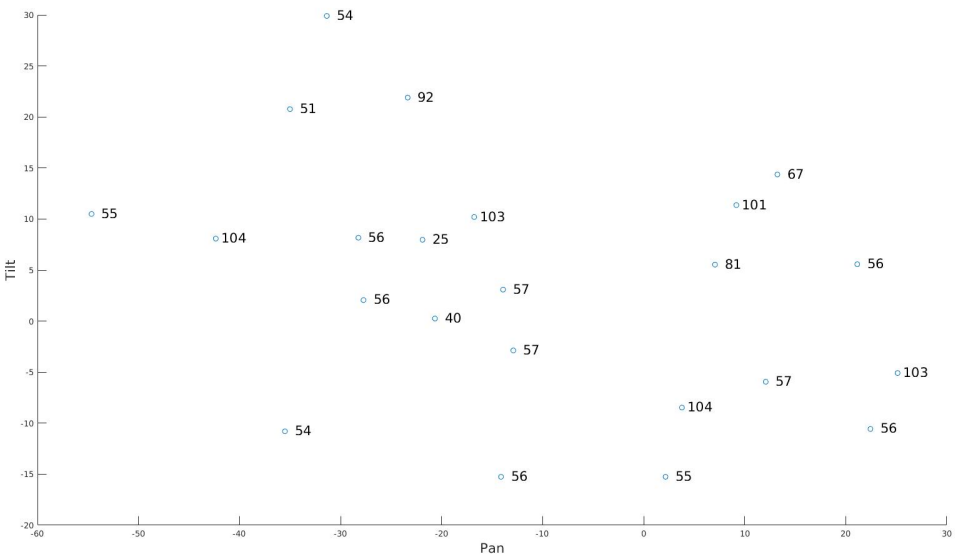
(a) Place 1 - Scene 2



(b) Place 1 - Scene 2 Map

Figure A.4. Visual and the object map of place 1 - scene 2

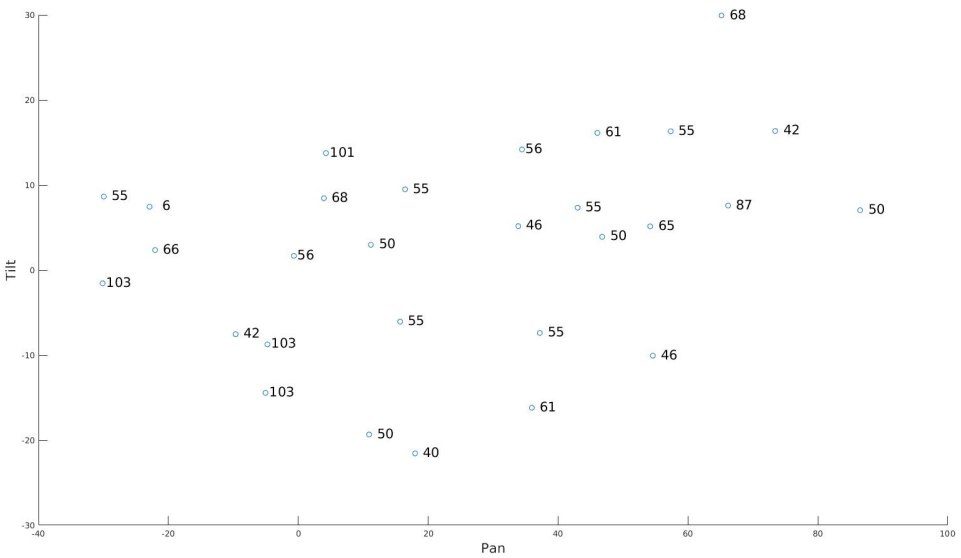
(a) Place 2 - Scene 1



(b) Place 2 - Scene 1 Map

Figure A.5. Visual and the object map of place 2 - visit 1

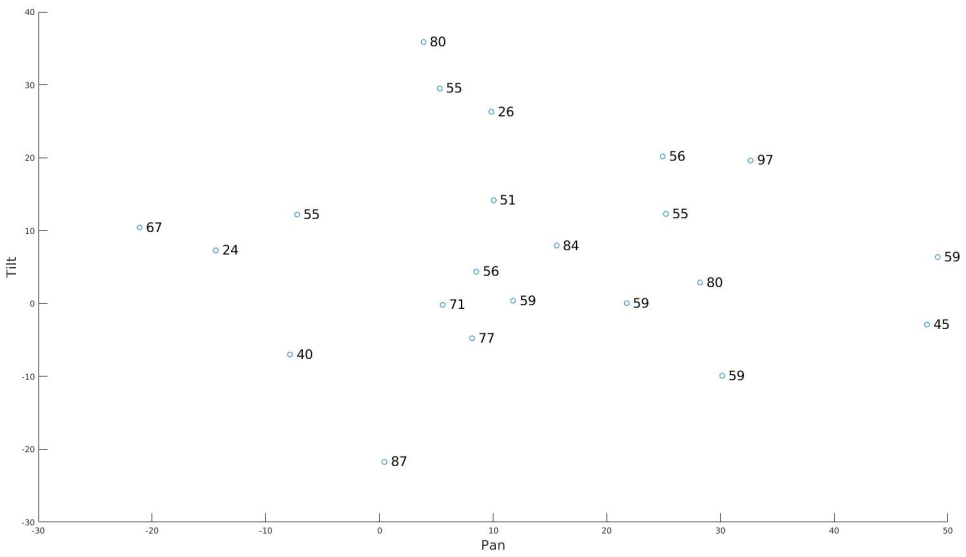
(a) Place 2 - visit 2



(b) Place 2 - Visit 2 Map

Figure A.6. Visual and the object map of place 2 - visit 2

(a) Place 3



(b) Place 3 Map

Figure A.7. Visual and the object map of place 3

APPENDIX B: HARDWARE & SOFTWARE

In this chapter, the hardware and software components of the robot are provided.

B.1. Hardware

The setup of the Kobuki turtlebot hardware is described. The robot is equipped with a pan-tilt RGB-D camera. The configuration space of the pan-tilt mechanism is $\mathcal{F} = [-90°, 90°] \times [-30°, 30°]$. The two motors are controlled through an arduino card. The camera is a Kinect having a 57.5° horizontal and 43.5° vertical field of view. The incoming images are of dimension $640x480$. Both the arduino and the camera are powered by the turtlebot's power output. As a processing unit, an ultrabook with 1.5Ghz is used. The electrical and electronic connections on the robot are as given in Figure B.1.

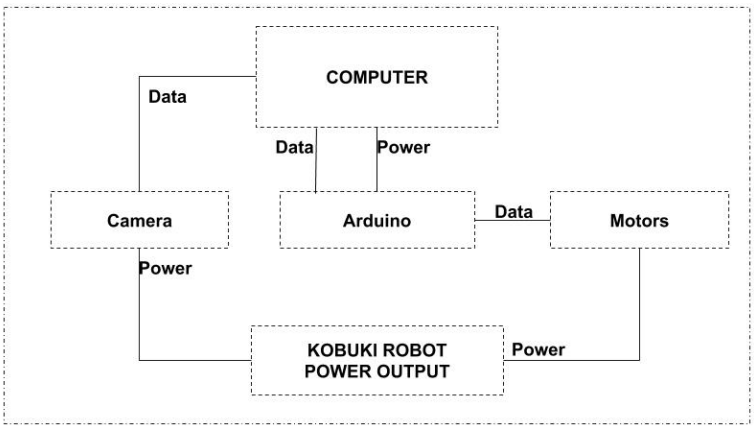


Figure B.1. Robot hardware connections

B.2. Programming Languages

The robot's software runs under the ROS framework [78]. Caffe [79] deep learning frame work is used for producing visual descriptors. Image processing is done by using Opencv computer vision library [80]. Additionally, Alglib numerical analysis and data processing library is used for the Ward's hierarchical clustering. The following software are required:

- Ubuntu 14.04
- ROS Indigo
- OpenCV which is included in ROS Indigo
- Caffe frame work
- Alglib library

Three different programming languages are used. The codes on the robot are C++. The CNN visual descriptors are written in python. Additionally, MATLAB is used for prototype implementations and data visualization.

B.3. Running Software

The software developed is located as follows:

- The codes are within the ros workspace which is located in the home folder. $defs.h$ is where the external parameters are given.
- $ObjectProcessor$ is a functional class which is used for constructing the function φ to control camera movements.
- $Segmenttrack$ class is a functional class where coherent segments are determined.
- $Graphmatch$ is a functional class used in $Segmenttrack$ to match segments from different time instance.

The robot can be made operational through executing the following commands:

(i) Connect Arduino and Camera to the Laptop.

(ii) In a terminal run *roscore*.

(iii) In a new terminal, run *rosrun apes_arduino apes_arduino_node*

(iv) In a new terminal, run *rosrun apes_head apes_head_node*

(v) In a new terminal, run *rosrun attentive_robot attentive_robot_node*

(vi) In a new terminal, run *roslaunch freenect_launch freenect.launch*