creative commons
COMMONS DEED

저작자표시-비영리-변경금지 2.0 대한민국

이용자는 아래의 조건을 따르는 경우에 한하여 자유롭게

- 이 저작물을 복제, 배포, 전송, 전시, 공연 및 방송할 수 있습니다.

다음과 같은 조건을 따라야 합니다:



저작자표시. 귀하는 원저작자를 표시하여야 합니다.



비영리. 귀하는 이 저작물을 영리 목적으로 이용할 수 없습니다.



변경금지. 귀하는 이 저작물을 개작, 변형 또는 가공할 수 없습니다.

- 귀하는, 이 저작물의 재이용이나 배포의 경우, 이 저작물에 적용된 이용허락조건을 명확하게 나타내어야 합니다.
- 저작권자로부터 별도의 허가를 받으면 이러한 조건들은 적용되지 않습니다.

저작권법에 따른 이용자의 권리는 위의 내용에 의하여 영향을 받지 않습니다.

이것은 이용허락규약(Legal Code)을 이해하기 쉽게 요약한 것입니다.

Disclaimer

dCollection

M.S. THESIS

Deep Learning Methods for Perception and Navigation of Service Robots

서비스 로봇의 지각과 자율 주행을 위한 딥 러닝 방법 연구

BY

Jinyoung Choi

FEBRUARY 2018

INTERDISCIPLINARY PROGRAM IN COGNITIVE SCIENCE
COLLEGE OF HUMANITIES
SEOUL NATIONAL UNIVERSITY

M.S. THESIS

Deep Learning Methods for Perception and Navigation of Service Robots

서비스 로봇의 지각과 자율 주행을 위한 딥 러닝 방법 연구

BY

Jinyoung Choi

FEBRUARY 2018

INTERDISCIPLINARY PROGRAM IN COGNITIVE SCIENCE
COLLEGE OF HUMANITIES
SEOUL NATIONAL UNIVERSITY

Deep Learning Methods for Perception and Navigation of Service Robots

서비스 로봇의 지각과 자율 주행을 위한 딥 러닝 방법 연구

지도교수 Byoung-Tak Zhang

이 논문을 공학석사 학위논문으로 제출함

2017 년 12 월

서울대학교 대학원

인지과학 협동과정

Jinyoung Choi

Jinyoung Choi의 공학석사 학위논문을 인준함

2017 년 12 월

위 원 장	김청택
부위원장	장병탁
위 원	김홍기

Abstract

To assist humans in everyday life, service robots require perception ability to understand the holistic context of the environment by integrating multimodal sensory information. Moreover, service robots have to be able to navigate freely in human environment to actually provide services to humans. Here we propose novel deep learning approaches to improve the perception and navigation performance of service robots. First, we introduce IPSRO (Integrated Perception for Service RObots) framework, which is ROS-friendly integrated perception system that we have recently open-sourced. IPSRO can flexibly integrate several perception modules including state-of-the-art deep learning models to extract rich and useful perceptual information from the environment based on a unified perception representation. On top of that, IPSRO can process the generated perceptual information to perform complex perception tasks. Second, we propose a NavNet, which is a deep reinforcement learning model for robust autonomous navigation. NavNet uses state-of-the-art off-policy deep reinforcement learning model to utilize previous experiences and human demonstration. We present the two system's capability evaluated in home-like environments. We also demonstrate the results of the deployment of IPSRO on the RoboCup@Home 2017 Social Standard Platform League.

Keywords: Mobile service robots, Deep learning, Reinforcement learning, Computer vision

Student Number: 2016-20100

Contents

Abstract **i**

Chapter 1 Introduction **1**

Chapter 2 Related Works **4**

2.1 Perception-Action Cycle of Cognitive Agent 4

2.2 Deep Learning Methods for Perception 5

2.3 Deep Learning Methods for Autonomous Navigation 8

Chapter 3 Learning to Perceive - IPSRO Framework **11**

3.1 Architecture 11

3.1.1 ROI Extraction 11

3.1.2 Tag Extraction 12

3.1.3 Information Integration 13

3.1.4 Reasoning Engine 14

3.2 Experiments 16

3.2.1 Memory Usage and Running Speed 16

3.2.2 Executing Complex Commands 17

3.3 Field Experiments 19

Chapter 4 Learning to Navigate - NavNet **23**

4.1 Preliminaries: Deep Reinforcement Learning 23

4.2 Architecture 24

4.3 Training 25

4.3.1 Gazebo Indoor Simulator Environment 25

4.3.2 Task Settings 26

4.3.3 Implementation Details 26

4.4 Experiments 27

Chapter 5 Conculsion **29**

초록 **37**

Acknowledgements **38**

List of Figures

Figure 1.1 SoftBank's Pepper robot equipped with IPSRO performing the Speech and Person Recognition task at RoboCup@Home 2017. 3

Figure 2.1 Deep learning approaches for various perception tasks (Top : Object detection, Middle : Image captioning, Bottom : Pose estimation. Figures are from [11], [14], and [16] respectively.) 7

Figure 2.2 Deep learning approaches for autonomous navigation (Top : Deep reinforcement learning, Bottom : Mapping and planning. Figures are from [22] and [26] respectively.) 10

Figure 3.1 Pipeline of IPSRO framework 12

Figure 3.2 GPU memory usage (blue) and running speed (orange) according to minibatch size 17

Figure 3.3 Home-like environment used in command executing experiments 19

Figure 3.4 Visualization of perceptual information in command executing experiments(Left: command1, Middle: command2, Right: command3. The raw images are shown in the top row and annotated images in the bottom.) 19
Figure 3.5 Pepper robot with IPSRO framework performing GPSR tasks in RoboCup@Home 2017 SSPL competition. (Left : raw images, Right : images annotated by IPSRO framework) 22

Figure 4.1 Network Architecture of NavNet 25
Figure 4.2 Environment used in navigation experiments 26
Figure 4.3 Learning curves in navigation experiments. (Top : comparison between all models, Bottom : comparison between NavNet with or without human demonstrations) . 28

List of Tables

Table 3.1 Score sheet for RoboCup@Home 2017 SSPL 21

Table 4.1 Hyper parameters for NavNet experiments 27

Chapter 1

Introduction

Service robots are an active topic in robotics research and development due to their potential abilities in everyday life. In the human environment, service robots can not only assist the elderly, handicapped people and children but also offer various physical or emotional support for ordinary users. They can also assist the workers and promote the businesses by actively interacting with customers or conducting tasks such as placing an order or providing guidance. These service robots are often expected to possess high level cognitive skills such as object manipulation, human-robot interaction (HRI), or socially-aware navigation. To perform these tasks, cognitive ability to acquire rich information about objects and people is essential. These robots should be able to recognize people and objects in everyday life. More importantly, they need to accurately recognize rich attributes about people such as their gesture, pose, gender, and identity. On top of that, service robots require navigation ability which enables the robots to follow, guide, or approach to human or to gather data about the environment actively. This perception-action cycle is fundamental part of the

service robots. Several frameworks such as OpenCV [1] or ROS navigation stack [2] are available for these tasks. However, most of the frameworks only provide limited functions and deploy traditional techniques such as HOG [3] or ORB [4]. Although deep learning-based methods have recently made breakthroughs to solve many problems in computer vision, natural language processing, and robot control, they have mostly focused on individual artificial tasks such as object detection, image captioning, or video games.

In this paper, we propose a novel framework for perception-action cycle of the service robots. First, we introduce IPSRO (Integrated Perception for Service RObots) framework, which is novel integrated perception system for service robots. Our ROS-friendly and open-sourced system provides an elastic parallel pipeline to integrate multimodal vision modules, including state-of-the-art deep learning models.

IPSRO extracts rich attributes of objects and people from the scene in real-time then builds perceptual information using a unified perception representation (Figure 1.1). IPSRO also provides a reasoning engine that utilizes the perceptual information to perform complex and difficult perception tasks.

Second, we propose a NavNet framework for robust navigation of the mobile robots. NavNet uses deep reinforcement learning method to learn autonomous navigation. Contrast to previous approach, NavNet uses off-policy deep reinforcement learning algorithm so it can incorporate all sorts of trajectories, including human demonstrations, in its learning process. This leads to faster and more stable learning than previous approach.

We first demonstrate our perception system's capability to perform sophisticated command execution tasks in the experiments. We also deployed IPSRO in the Robocup@Home 2017 Social Standard Platform League (SSPL), the competition to test the integrated cognitive ability of social service robots in a

realistic home environment. The Softbank Pepper robot equipped with IPSRO showed state-of-the-art performance in all cognitive tests including General Purpose Service Robot (GPSR) task and won the first place in the challenge by a large margin. Secondly, we test the capability of the NavNet architecture in the simulated 3D environments. NavNet showed better performance than baseline models in challenging indoor environment by efficiently exploring the environment and utilizing its ability to incorporate human demonstrations.

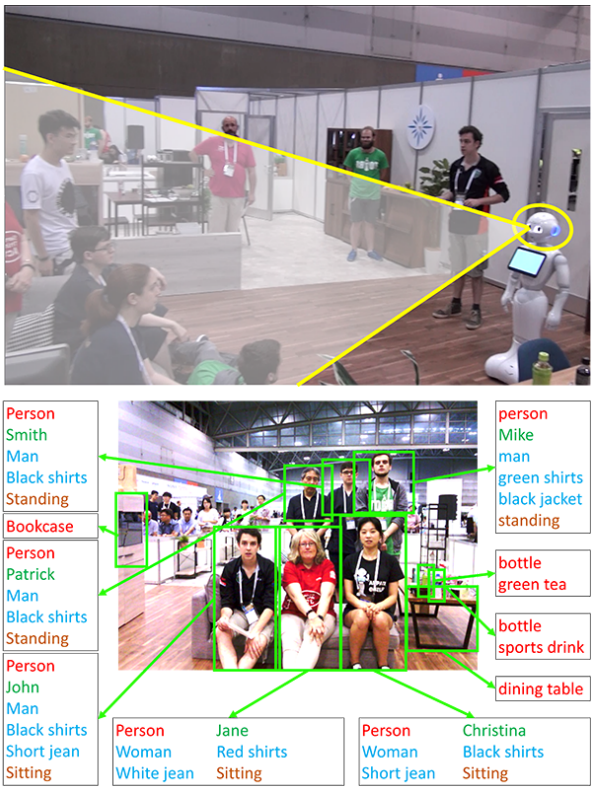


Figure 1.1 SoftBank's Pepper robot equipped with IPSRO performing the Speech and Person Recognition task at RoboCup@Home 2017.

Chapter 2

Related Works

2.1 Perception-Action Cycle of Cognitive Agent

In the cognitive robotics, the perception-action cycle is considered as essential part of the cognitive agents. The perception-action cycle is the circular flow of information that takes place between the agent and its environment in the course of a sensory-guided sequence of behavior towards a goal [5]. The perception is a ability to extract useful information from the unstructured sensory inputs. The result of the perception is then used to control the action, which is the ability to influence the environment to achieve the agent's goal. The result of the action is used to update the perception by exploiting the changes or reveal new states of the environment. This cycle gives robots the ability to actively seek information relevant to its goal and perform appropriate actions to achieve the goal. In this chapter, we review the deep learning methods for perception and action of the service robots. For actions, we focus on the navigation ability of the service robots.

2.2 Deep Learning Methods for Perception

To interact with or assist people, service robots require a perception framework that can provide information such as the location/type of objects and the identity/pose/gender of people in the environment.

Many perception frameworks have been used in service robots. OpenCV or OpenNI have been widely used to perform perception tasks such as object detection, human pose estimation. The Strands framework [6] provides an integrated system for people perception. Most of these frameworks focus on only a few tasks such as object detection or face recognition. Furthermore, these frameworks use traditional vision methods that are known to be vulnerable to illumination change or translation of objects. Those frameworks also lack a reasoning engine that can build perceptual information and reason about it. Frameworks such as RoboSherlock [7][8] provide sophisticated reasoning engines on top of the integrated perception pipeline but they focus only on object manipulation and they also use traditional vision modules. These limitations in perception frameworks often limit service robots to only show good performance in well-defined tasks in a controlled environment.

Recently, following the remarkable success of deep learning in object recognition [9], many deep learning based perception models have been proposed (Figure 2.1). Deep learning based approaches are known to be robust to illumination or translation and have marked state-of-the-art performances in many vision tasks such as object detection [10][11][12], image description [13][14], and pose estimation [15][16].

These models show superior performance than more traditional approaches. However, these models are not enough to be deployed in complex and realistic perception tasks since they mostly focus on individual tasks such as object

detection, face detection, or object recognition. Furthermore, these models also lack reasoning engines that can process perceptual information efficiently.

To overcome these limitations, our IPSRO framework provides an efficient integrated perception pipeline that can integrate many perception modules including state-of-the-art deep learning models. We also provide a reasoning engine that can build perceptual information from the output of the pipeline and perform complex perception tasks.

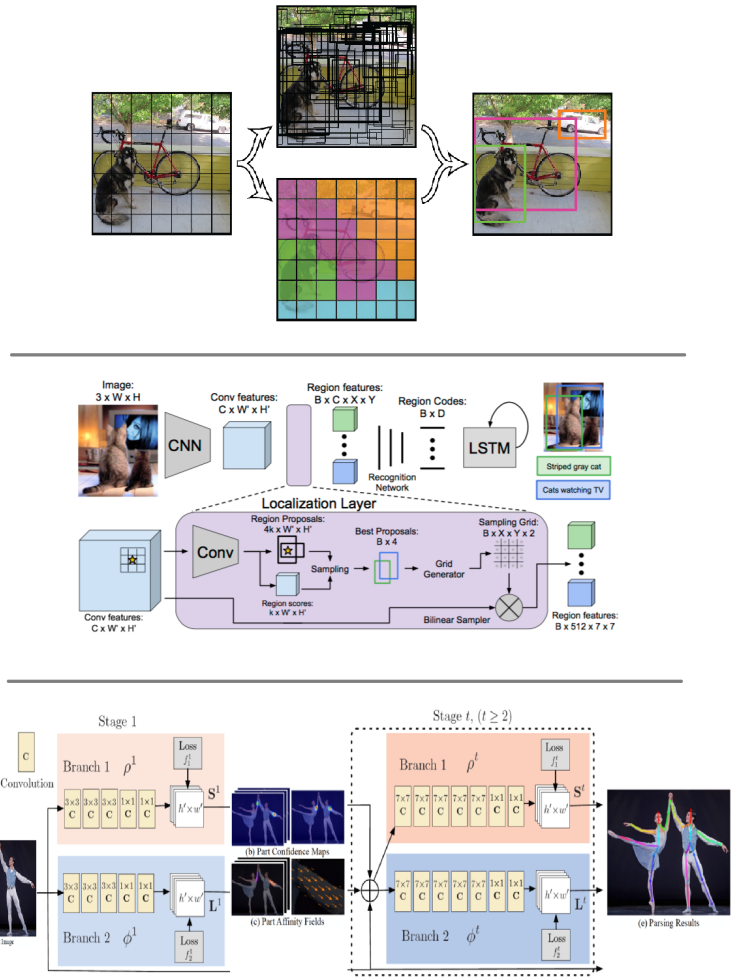


Figure 2.1 Deep learning approaches for various perception tasks (Top : Object detection, Middle : Image captioning, Bottom : Pose estimation. Figures are from [11], [14], and [16] respectively.)

2.3 Deep Learning Methods for Autonomous Navigation

Autonomous navigation is crucial ability for service robots since they have to move freely in the human environment to interact or assist humans.

For this task, simultaneous mapping and localization (SLAM) algorithms such as [17], [18], and [19] have been used to build the map of the environment. The robots used built map to plan its path and navigate using planning algorithms such as [20] and [21]. This traditional approaches show low performance without expensive equipment such as laser sensor. Furthermore, these approaches are prone to environment changes so the user has to do mapping whenever the environment changes. To overcome these shortcomings of the traditional algorithms, visual navigation using the deep learning is actively studied (Figure 2.2).

Recently proposed method [22] used deep reinforcement learning for the visual navigation. The network processes current observation image and the target image using the Siamese convolutional neural network [23]. Then the concatenated representations are used to estimate the value function and the policy distribution. Although this approach showed some degree of generalization across the target locations and the environment changes, it requires the 3D reconstruction of the environment to be deployed in the real world and has to be fine-tuned with millions of samples.

Works such as [24], [25], and [26] used memory-augmented neural network to tackle this problem.

[24] used Memory Network architecture and attention mechanism to store previous observations. This architecture showed good performance in Minecraft maze solving problems but limited memory slots are not appropriate for visual

navigation tasks, which have multiple, changing goals.

[25] showed better results in VisDoom [27] environment using 2D memory to construct the map of the environment. However, it requires exact location of the agent in the environment, which is intractable in the real world.

[24] builds ego-centric map representation using the action dependent affine transform and value iteration network [28]. This method showed good performance in the THOR AI2 indoor dataset [22]. However, it can only maintain the ego-centric map and requires action labels, which are expensive to acquire.

Contrast to previous works, we use state-of-the-art off-policy deep reinforcement learning to utilize human demonstrations.

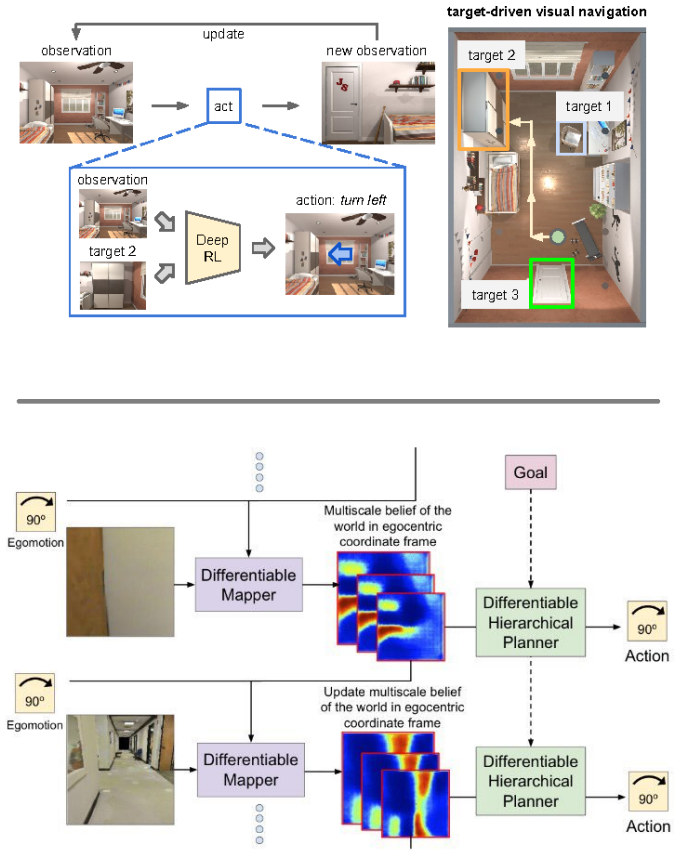


Figure 2.2 Deep learning approaches for autonomous navigation (Top : Deep reinforcement learning, Bottom : Mapping and planning. Figures are from [22] and [26] respectively.)

Chapter 3

Learning to Perceive - IPSRO Framework

3.1 Architecture

The perception pipeline of IPSRO consists of 3 stages (Figure 3.1): ROI segmentation, attribute extraction, perceptual information building. The output from the pipeline is then processed by the reasoning engine. We will cover each stage in detail in following subsections.

3.1.1 ROI Extraction

To extract the perceptual information from scene images, the system must determine the region of interests (ROIs). For any object detector or semantic segmentation models can be used alone or together to figure out the ROIs in the scene. One can use the state-of-art models such as YOLO [11], OpenFace [29], or Mask R-CNN [30]. By default, we used YOLO [11] object detector in this stage.

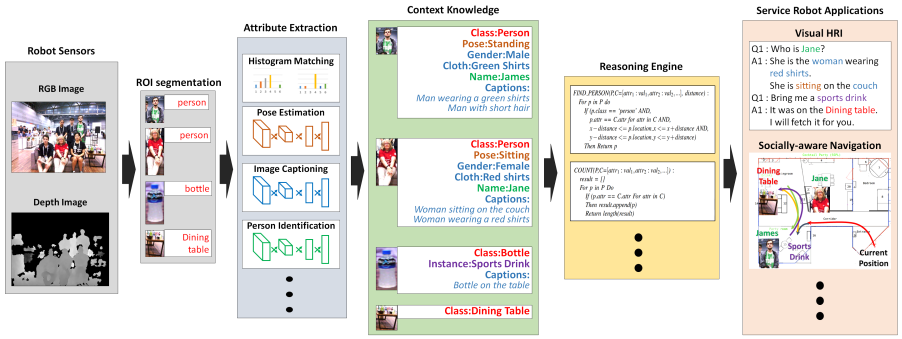


Figure 3.1 Pipeline of IPSRO framework

3.1.2 Tag Extraction

Service robots require more sophisticated attributes about objects and people to perform complex cognitive tasks. It can be realized by using expert modules based on advanced machine learning for specific tasks, such as posture or gesture recognition, people identification, and objects localization in the map.

In the attribute extraction stage, expert modules such as the pose estimation module and image captioning module can process the segmented ROIs to extract more sophisticated attributes. Expert modules can be deployed in a parallel or sequential manner depending on type of the information the robot requires in the environment. IPSRO supports efficient deployment of several deep learning models by building a mini-batch of ROIs and processing them in parallel.

However, the GPU memory can be insufficient even in high-end computers when one deploys several deep learning models in parallel. To alleviate this problem, we provide the mini-batch size restriction. If the size of the mini-batch is too large, IPSRO splits the mini-batch into small chunks and processes them sequentially. This restriction slows down the system when there is a large number of ROIs but enables IPSRO to run with low cost GPU systems. We

demonstrate the effect of this restriction in Section 3.2.1.

By default, we deploy deep learning modules for human pose estimation [16], image captioning [14], and person re-identification [31]. We also deploy the location estimation module that estimates the location of the ROI using point clouds, and the histogram matching module that compares the color histograms of ROIs to the stored images to classify fine-grained object instances without training. Other modules such as the fine-grained image classifier [9] can be easily included in the pipeline.

One can use different languages (C, Python, Matlab) or APIs (OpenCV, TensorFlow, PyTorch, etc) for each expert module as far as the language is ROS-compatible.

3.1.3 Information Integration

The perceptual information building module aggregates all attributes extracted by the expert modules. First, it transforms the attributes extracted from each expert modules into a unified attribute representation.

For example, the pose estimation module's output is processed to '($pose$: $sitting$)', the image captioning model's output is processed to '($description$: $man\ wearing\ a\ black\ shirt$)', and the location estimation module's output is processed to '($location_x$: 0.45)', '($location_y$: 2.2)'.

Then, the module aggregates these attributes to build the perceptual information.

Since IPSRO is able to execute expert modules in parallel, contradictory annotations can be generated for the same attribute. IPSRO can utilize the confidence values from the expert modules when the confidence values are available. One can also create the pipeline sequentially to solve the contradictory annotation problem. The output of the perception pipeline can be stored and

be used for more sophisticated reasoning that involves using past contexts.

3.1.4 Reasoning Engine

reasoning engine module performs reasoning about integrated perceptual information to carry out sophisticated perception tasks. Functions for reasoning can be defined by specifying rules regarding to the attributes. These functions can be generally expressed as Algorithm 1.

Algorithm 1: Reasoning Process in IPSRO

```
Result: result R
perceptual_information P
conditions C
attributes_to_retrieve A
for instance p in P do
    if attributes of p satisfies C then
        R.append( A of p )
    end
end
```

In Algorithm 1, perceptual information (P) can be the current perceptual information or the history of previous information. Conditions (C) can be conditions regarding to attributes such as '$class == person$' or '$0.3 <= location_x <= 0.6$'. Attributes to retrieve (A) can be the set of attributes to retrieve, such as '*name*' or '*description*'.

One can define functions to do specific perception tasks. For example, the function *FIND_PEOPLE* returns the people who have specific attribute values in a specific area using the following pseudo-codes.

FIND_PEOPLE(P,C=$[attr_1 : val_1, attr_2 : val_2, ...]$*, distance)* :

result = []

For p in P do

If (p.class == 'person' And

$[p.attr == C.attr$ *for attr in C] And*

$x - distance <= p.location_x <= x + distance$ *And*

$y - distance <= p.location_y <= y + distance)$

Then result.append(p)

Return result

where P is perceptual information from the perception pipeline, and C is the conditions to find. The function $COUNT$ returns the number of objects with specific attributes using following pseudo-codes.

COUNT(P,C=$[attr_1 : val_1, attr_2 : val_2, ...]$*)* :

result = []

For p in P do

If [p.attr == C.attr For attr in C]

Then result.append(p)

Return length(result)

These kinds of functions can be used for various perception tasks such as retrieving a past trajectory of the specific person, detecting a person who needs assistance, or describing an unknown person in a specific area.

Combined with navigation or dialog systems, IPSRO can be used in various cognitive tasks such as socially-aware navigation, visual HRI, and life pattern learning in everyday lives.

3.2 Experiments

We tested IPSRO on various conditions and evaluated its capability to aggregate many modules. We also measured IPSRO's performance on complex perception tasks.

3.2.1 Memory Usage and Running Speed

We tested IPSRO which contained 6 modules in the pipeline. Among the 6 modules, 4 modules were state-of-the-art deep learning models that were each trained for different perception tasks. We used the YOLO [11] object detector module that was trained using TensorFlow [32] in the ROI segmentation stage. In the attribute extraction stage, we used the OpenPose [16] human pose estimation module, DenseCap [14] image caption generation module, and Person re-identification [31] module in parallel. These modules were trained using Pytorch, Torch and Keras, respectively. We also deployed a point-cloud-based location estimator and color histogram matching module in our pipeline. We tested the system using the Softbank Pepper platform and external GPU server with an i7-octa core CPU, Nvidia Geforce TitanX GPU, 32 GB RAM. To test the scalability of IPSRO, we evaluated the GPU memory usage and running speed with various mini-batch sizes (Figure 3.2). As seen in Figure 3.2, IPSRO showed real-time processing speed. On top of that, IPSRO could maintain stable GPU memory usage due to the mini-batch size restriction. Although this restriction slowed down the system when the mini-batch size was larger than 120, IPSRO maintained real-time processing speed while running 4 heavy deep learning modules.

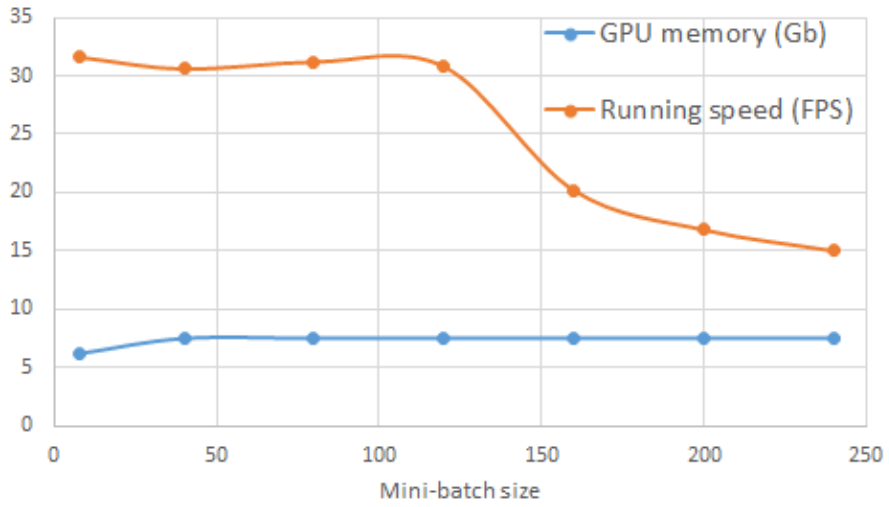


Figure 3.2 GPU memory usage (blue) and running speed (orange) according to minibatch size

3.2.2 Executing Complex Commands

We tested the capability of IPSRO to integrate and reason about the perceptual information by making the robot execute several complex commands in a home-like environment (Figure 3.3). The commands contained several sophisticated perception tasks. We used the perception pipeline described in Section3.2.1 for all experiments.

Find James in the kitchen and guide him to entrance

To perform this task, the robot has to complete 2 perception tasks. First, locate James in the kitchen and second locate James while guiding him to the entrance, ensuring that he is following the robot. These perception tasks can be done using the *FIND_PEOPLE* function defined in Section **??**. The robot moves to

the kitchen then calls "*target = FIND_PEOPLE(P, C={'name':'James', 'location_x':x_kitchen, 'location_y':y_kitchen}, distance=1.0)*" to find James. Attribute 'name' is extracted by the person re-identification module and 'location' is extracted by the point-cloud based location estimation module. After finding James, the robot navigates to the entrance. While navigating, the robot turns 180 degrees every 20 seconds and calls "*target = FIND_PEOPLE(P, C={'name':'James', 'location_x':x_robot, 'location_y':y_robot}, distance=2.0)*" to ensure that James is within 2 meters from the robot.

IPSRO successfully recognized James and succeeded to execute the command (Figure 3.4, left).

Tell me how many coke bottles are on the table

In this task, the robot has to count the number of specific kind of objects. We used the *COUNT* function defined in Section **??** as follows: "*number = COUNT(P, C={'class':'bottle', 'instance':'coke'})*". Attribute 'class' is extracted by the object detection module and 'instance' is extracted by the histogram matching module. IPSRO successfully recognized 2 coke bottles (Figure 3.4, middle).

Describe unknown people at the living room

In this task, 3 people stand in the living room and 2 of them are unknown to the robot. The robot has to find the people whose names cannot be identified and return descriptions of them. We used 'FIND_PEOPLE' function to do this task as follows: "*targets = FIND_PEOPLE(P, C={'name':'unknown', 'location_x':x_livingroom, 'location_y':y_livingroom}, distance=1.0)*" and "*descriptions = [t.description For t in targets]*". The attribute 'description' is extracted from the image captioning module. Our robot successfully recognized 2 un-

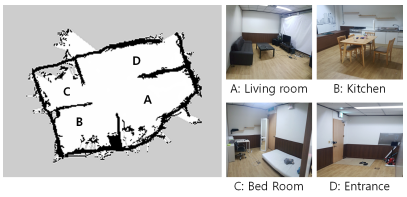


Figure 3.3 Home-like environment used in command executing experiments

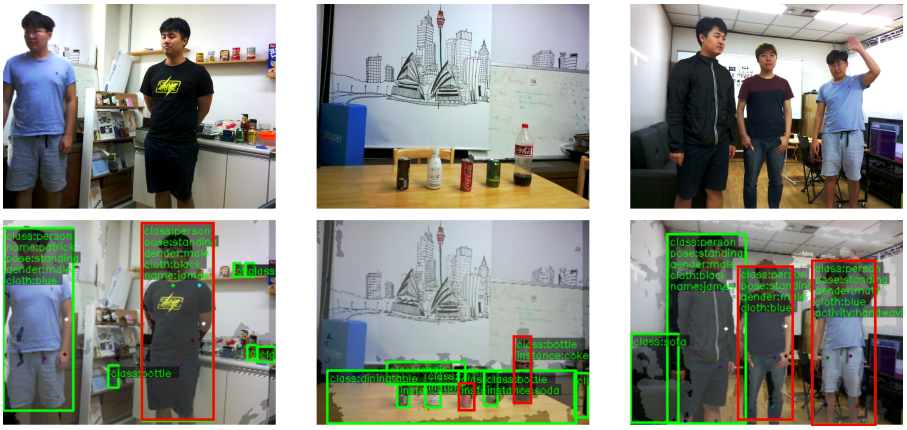


Figure 3.4 Visualization of perceptual information in command executing experiments(Left: command1, Middle: command2, Right: command3. The raw images are shown in the top row and annotated images in the bottom.)

known people (Figure 3.4, right). It described one unknown person standing in the middle, as a 'man wearing a blue shirt, pair of black shoes' and a second person standing on the right, as a 'man wearing a blue shirt, pair of blue jeans'.

3.3 Field Experiments

RoboCup@Home is known as the biggest competition for domestic service robots. The competition aims to evaluate robots' abilities and performances in a realistic non-standardized home environment [33]. To show IPSRO's per-

formance, we participated in the Social Standard Platform League (SSPL) of the RoboCup@Home 2017, which used 'Pepper' of Softbank Robotics as the standard robot platform. We used an external computing server equipped with a GPU to run IPSRO. Then we deployed the robot to perform 7 scenarios in the competition. The arena was set up as a pseudo-home environment. Because the arena was not opened before the competition, it was important to generalize robot's abilities to unknown situations. We demonstrate IPSRO's performance in the General Purpose Service Robot (GPSR) scenario, which was the most difficult task among the 7 scenarios. For details about the competition, please refer to the rulebook [34]. In the GPSR task, the robot has to execute arbitrary voice commands. The commands include but are not limited to finding an object or person, answering a question, following or guiding a person, counting the number of objects, and describing a person or place. In the competition, our robot successfully performed the following 3 commands and scored highest scores among 7 teams (Figure 3.5).

- Say the time to Jacob at the kitchen table
- Find a hair spray at the kitchen table
- Tell me how many cokes are on the desk

We also applied IPSRO to all the other tasks in the competition. Due to rich perception, our robot could accomplish first place in all 7 scenarios by a large margin (Table 3.1).

The videos of the competitions can be found in `https://youtu.be/166IugGGhN8`

Team	Speech Recognition	Cocktail Party	Help Me Carry	GPSR	Restaurant	EEGPSR	Tour Guide
AUPAIR (ours)	**117.5**	**30**	**10**	**42.5**	**40**	**70**	**95**
UTS unleashed	85.5	27.5	0	17.5	0	0	0
SPQReL	32.5	10	5	7.5	10	20	0
Kame Rider	60	0	0	0	15	0	0
UChile Peppers	50	0	0	0	0	0	0
UvA Home	47.5	0	0	0	0	0	0
ToBI Pepper	17.5	7.5	0	0	0	0	0

Table 3.1 Score sheet for RoboCup@Home 2017 SSPL



Figure 3.5 Pepper robot with IPSRO framework performing GPSR tasks in RoboCup@Home 2017 SSPL competition. (Left : raw images, Right : images annotated by IPSRO framework)

Chapter 4

Learning to Navigate - NavNet

4.1 Preliminaries: Deep Reinforcement Learning

Reinforcement learning is the type of the machine learning that aims to maximize the accumulated discounted reward (value function) of the agent. The agent has to learn behavior policy to interact with the environment in advantageous way.

This can be formulated by Bellman equation :

$$Q(s_t, a_t) = r_t + \gamma * Q(s_{t+1}, a_{t+1})$$

where s_t is the observation of the environment in time step t, a_t is the agent's action in step t, r_t is the reward in step t, γ is the discount coefficient, and $Q(s_t, a_t)$ is the value function in time step t.

Optimization of the policy can be done by either performing direct estimation on the value function, or using the gradient ascent algorithms to the parameterized policy distribution.

Deep reinforcement learning (DRL) is reinforcement learning algorithms that uses deep neural network to estimate the value function [35] or parameterize the policy [36].

DRL is actively studied area because it can abstract the vast state-action space automatically and shows high performance on difficult control tasks such as video games [35] or robot control [22].

In this paper,we deploy DRL algorithms for visual navigation tasks.

4.2 Architecture

Our architecture is based on [22]. We used Siamese convolutional neural network [23] to extract the representation of the current observation and goal image. We changed A3C [36] algorithm of [22] to DQN [35] algorithm. Since DQN is the off-policy algorithm, we can use all previous experience and more importantly, human demonstrations. We also deployed recent advances in DQN algorithms : Double DQN [37], Deuling network [38], Prioritized experience replay [39], Noisy network [40].

The network architecture we used in the experiments are illustrated in Figure 4.1. The convolutional layers are consist of 4 layers. Each convolutional layer has 32 filters of size 3 and stride 2. For the last convolutional layer, we used 256 filters. Then we used global average pooling and 2 fully connected layers with 256 units to extract visual features. Same parameters are used to extract features of current observation and goal image. Two features are concatenated into 512-dimension vector and used to extract state value and advantages (state value and policy in A3C baseline [22]). We used Exponential linear unit [41] for non-linear activation function.

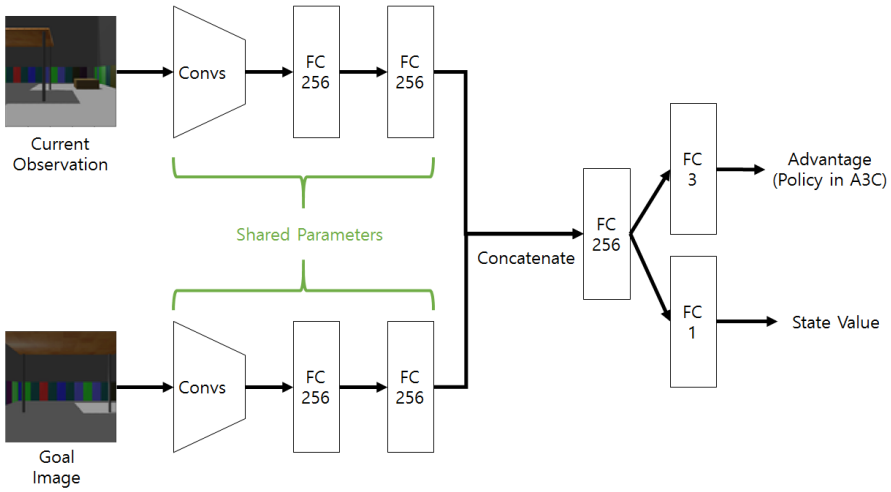


Figure 4.1 Network Architecture of NavNet

4.3 Training

4.3.1 Gazebo Indoor Simulator Environment

Training a deep reinforcement learning model on the real robot is very challenging since the real robots require a lot of time and energy to acquire the experience. Therefore, we built the indoor simulator using the Gazebo simulator (`http://gazebosim.org/`). The simulator procedurally generates the indoor environments by placing the corridors, rooms, and objects. The number of corridors, rooms, objects are sampled from pre-defined probability distributions and location of them are uniformly sampled in the map. The samples of generated indoor environments are presented in Figure 4.2. The generated environments are converted into grid-world like data then are used for training.

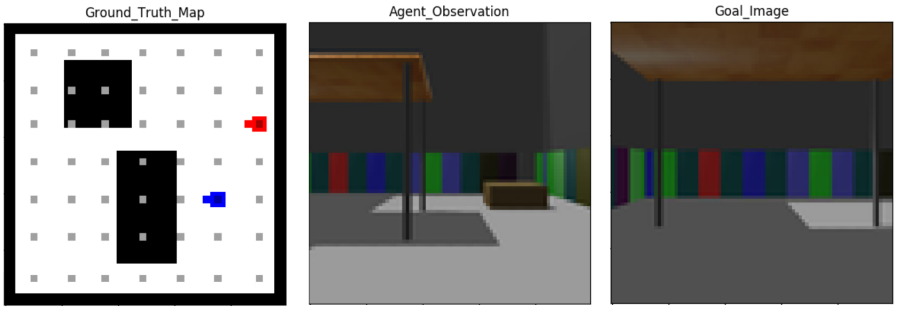


Figure 4.2 Environment used in navigation experiments

4.3.2 Task Settings

In the grid-world like environment, the agent observes the first-person view image of current location. Also, the first-person view from the goal location is given to the agent. The goal of training is reaching the goal using two images. In each time step, The agent can move 40cm forward or turn 90 degree (left or right). The agent receives the reward of 10.0 when it reaches the goal. To encourage the agent to find shortest path, the reward of -0.01 is given in every time step. Also, to prevent the agent collide to the wall or objects, the reward of -0.05 is given when the agent collides to wall or object. At the beginning of each episode, the goal location and agent location is changed randomly. Each episode in the training session is limited to 200 time steps.

4.3.3 Implementation Details

We used Pytorch (`http://pytorch.org/`) to implement the networks. Hyper parameters used in experiments are presented in Table 4.1

Parameter	Value
mini batch size	64
learning interval (steps)	4
learning start step (steps)	10,000
size of replay memory	100,000
optimizer & learning rate	Adam , 0.0001
discount	0.99
α in prioritized replay	0.5
multi-step returns	5
noisy network variance	0.5
entropy regularization coefficient (A3C)	0.01
number of workers (A3C)	8
gradient clipping	40

Table 4.1 Hyper parameters for NavNet experiments

4.4 Experiments

We trained our model and baseline [22] in the one of generated environment. To show the effect of the human demonstration, our model is trained with and without 1,000 time steps of human demonstrations. We replaced 50% of mini batch to the human demonstrations as proposed in [42]. The learning curve of models are presented in Figure 4.3.

As seen in the Figure 4.3 (top), our model learned much more faster and showed better performance than the A3C baseline with or without human demonstrations. We hypothesize that on-policy exploration of the A3C algorithm prevented the agent from crossing narrow pathway due to the minus rewards from collision.

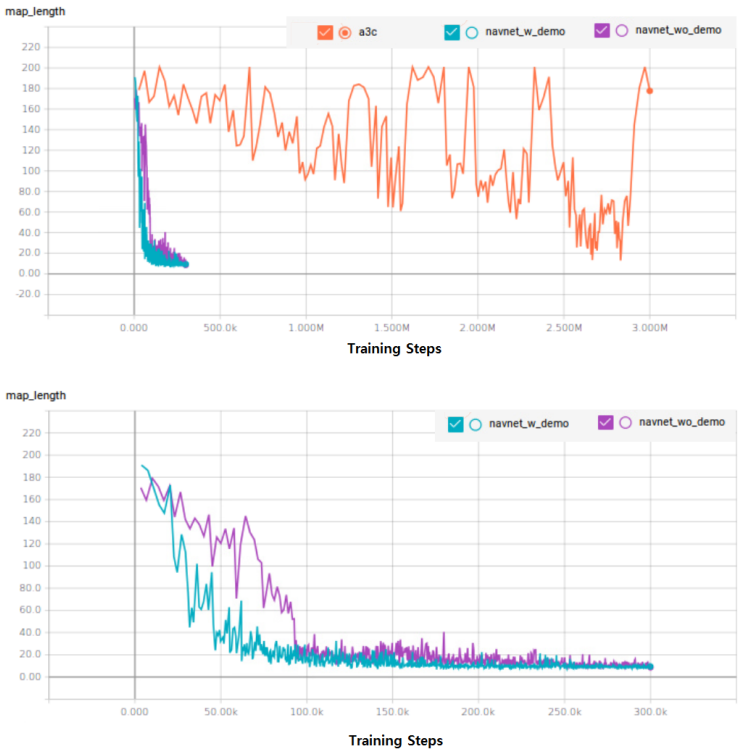


Figure 4.3 Learning curves in navigation experiments. (Top : comparison between all models, Bottom : comparison between NavNet with or without human demonstrations)

Between our models, model with human demonstration showed faster and more stable learning as we expected (Figure 4.3 (Bottom)).

Chapter 5

Conculsion

We propose deep learning frameworks for perception and navigation ability of the service robots. The perception framework IPSRO (Integrated Perception for Service RObots), which is ROS-based and open-sourced, can detect the Region of Interests (ROIs) from the scene by combining various algorithms. Rich and useful attributes can be extracted from the ROIs using many expert modules such as the pose estimation module or image captioning module. Then IPSRO integrates the information from the modules to build perceptual information. Perceptual information can be used to perform reasoning for complex perception tasks. We showed IPSRO's performance in complex command executing experiments and the RoboCup@Home 2017 SSPL competition. IPSRO succeeded to perform sophisticated perception tasks in both experiments.

The navigation framework, NavNet, uses state-of-the-art off-policy deep reinforcement learning algorithm to learn the autonomous navigation using only RGB images of current observation and goal. NavNet showed faster learning and higher performance than the baseline model in simulated indoor environ-

ment. It is also able to incorporate human demonstration to further accelerate the learning.

Based on the results of this work, our future work would be the improvement of proposed frameworks. Although IPSRO framework showed promising results in experiments, it relies on the rule-based reasoning engine. We expect that learning the reasoning process using extracted tags and query sentence would improve the generalization power and overall performance of the reasoning engine. Also, NavNet framework lacks the episodic memory that enables it to quickly adapt to the new environment. We plan to explore the efficient way to incorporate the episodic memory.

Finally, we studied perception ability and navigation ability independently in this work. We believe that merging the two framework will significantly improve the overall performance of the general purpose service robots.

Bibliography

[1] Gary Bradski. "The OpenCV Library." In: *Dr. Dobb's Journal: Software Tools for the Professional Programmer* 25.11 (2000), pp. 120–123.

[2] Morgan Quigley et al. "ROS: an open-source Robot Operating System". In: *ICRA workshop on open source software*. Vol. 3. 3.2. Kobe. 2009, p. 5.

[3] Navneet Dalal and Bill Triggs. "Histograms of oriented gradients for human detection". In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE. 2005, pp. 886–893.

[4] Ethan Rublee et al. "ORB: An efficient alternative to SIFT or SURF". In: *Computer Vision (ICCV), 2011 IEEE international conference on*. IEEE. 2011, pp. 2564–2571.

[5] Vassilis Cutsuridis, Amir Hussain, and John G Taylor. *Perception-Action Cycle: Models, Architectures, and Hardware*. Springer Science & Business Media, 2011.

[6] Christian Dondrup et al. "Real-time multisensor people tracking for human-robot spatial interaction". In: *Workshop on Machine Learning for So-*

cial Robotics at International Conference on Robotics and Automation (ICRA). ICRA/IEEE, 2015.

[7] Michael Beetz et al. "Robosherlock: Unstructured information processing for robot perception". In: *Robotics and Automation (ICRA), 2015 IEEE International Conference on*. IEEE. 2015, pp. 1549–1556.

[8] Ferenc Bálint-Benczédi, Patrick Mania, and Michael Beetz. "Scaling perception towards autonomous object manipulation in knowledge lies the power". In: *Robotics and Automation (ICRA), 2016 IEEE International Conference on*. IEEE. 2016, pp. 5774–5781.

[9] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in Neural Information Processing Systems*. 2012, pp. 1097–1105.

[10] Shaoqing Ren et al. "Faster R-CNN: Towards real-time object detection with region proposal networks". In: *Advances in Neural Information Processing Systems*. 2015, pp. 91–99.

[11] Joseph Redmon and Ali Farhadi. "YOLO9000: better, faster, stronger". In: *arXiv preprint arXiv:1612.08242* (2016).

[12] Wei Liu et al. "Ssd: Single shot multibox detector". In: *European Conference on Computer Vision*. Springer. 2016, pp. 21–37.

[13] Oriol Vinyals et al. "Show and tell: A neural image caption generator". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3156–3164.

[14] Justin Johnson, Andrej Karpathy, and Li Fei-Fei. "Densecap: Fully convolutional localization networks for dense captioning". In: *Proceedings of*

the IEEE Conference on Computer Vision and Pattern Recognition. 2016, pp. 4565–4574.

[15] Alexander Toshev and Christian Szegedy. "Deeppose: Human pose estimation via deep neural networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2014, pp. 1653–1660.

[16] Zhe Cao et al. "Realtime Multi-Person 2D Pose Estimation using Part Affinity Fields". In: *CVPR*. 2017.

[17] Bruno Steux and Oussama El Hamzaoui. "tinySLAM: A SLAM algorithm in less than 200 lines C-language program". In: *Control Automation Robotics & Vision (ICARCV), 2010 11th International Conference on*. IEEE. 2010, pp. 1975–1979.

[18] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. "ORB-SLAM: a versatile and accurate monocular SLAM system". In: *IEEE Transactions on Robotics* 31.5 (2015), pp. 1147–1163.

[19] Nikolas Engelhard et al. "Real-time 3D visual SLAM with a hand-held RGB-D camera". In: *Proc. of the RGB-D Workshop on 3D Perception in Robotics at the European Robotics Forum, Vasteras, Sweden*. Vol. 180. 2011, pp. 1–15.

[20] Steven M LaValle. "Rapidly-exploring random trees: A new tool for path planning". In: (1998).

[21] Sertac Karaman and Emilio Frazzoli. "Sampling-based algorithms for optimal motion planning". In: *The international journal of robotics research* 30.7 (2011), pp. 846–894.

[22] Yuke Zhu et al. “Target-driven visual navigation in indoor scenes using deep reinforcement learning”. In: *arXiv preprint arXiv:1609.05143* (2016).

[23] Sumit Chopra, Raia Hadsell, and Yann LeCun. “Learning a similarity metric discriminatively, with application to face verification”. In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 1. IEEE. 2005, pp. 539–546.

[24] Junhyuk Oh et al. “Control of memory, active perception, and action in minecraft”. In: *arXiv preprint arXiv:1605.09128* (2016).

[25] Emilio Parisotto and Ruslan Salakhutdinov. “Neural map: Structured memory for deep reinforcement learning”. In: *arXiv preprint arXiv:1702.08360* (2017).

[26] Saurabh Gupta et al. “Cognitive mapping and planning for visual navigation”. In: *arXiv preprint arXiv:1702.03920* (2017).

[27] Michał Kempka et al. “Vizdoom: A doom-based ai research platform for visual reinforcement learning”. In: *Computational Intelligence and Games (CIG), 2016 IEEE Conference on*. IEEE. 2016, pp. 1–8.

[28] Aviv Tamar et al. “Value iteration networks”. In: *Advances in Neural Information Processing Systems*. 2016, pp. 2154–2162.

[29] Tadas Baltrušaitis, Peter Robinson, and Louis-Philippe Morency. “Openface: an open source facial behavior analysis toolkit”. In: *Applications of Computer Vision (WACV), 2016 IEEE Winter Conference on*. IEEE. 2016, pp. 1–10.

[30] Kaiming He et al. “Mask r-cnn”. In: *arXiv preprint arXiv:1703.06870* (2017).

[31] Ejaz Ahmed, Michael Jones, and Tim K Marks. "An improved deep learning architecture for person re-identification". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3908–3916.

[32] Martın Abadi et al. "Tensorflow: Large-scale machine learning on heterogeneous distributed systems". In: *arXiv preprint arXiv:1603.04467* (2016).

[33] Luca Iocchi et al. "RoboCup@ Home: Analysis and results of evolving competitions for domestic and service robots". In: *Artificial Intelligence* 229 (2015), pp. 258–281.

[34] Beek Loy van et al. *RoboCup@Home 2017: Rules and Regulations*. 2017. URL: `http://www.robocupathome.org/rules/2017_rulebook.pdf`.

[35] Volodymyr Mnih et al. "Human-level control through deep reinforcement learning". In: *Nature* 518.7540 (2015), pp. 529–533.

[36] Volodymyr Mnih et al. "Asynchronous methods for deep reinforcement learning". In: *International Conference on Machine Learning*. 2016, pp. 1928–1937.

[37] Hado Van Hasselt, Arthur Guez, and David Silver. "Deep Reinforcement Learning with Double Q-Learning." In: *AAAI*. 2016, pp. 2094–2100.

[38] Ziyu Wang et al. "Dueling network architectures for deep reinforcement learning". In: *arXiv preprint arXiv:1511.06581* (2015).

[39] Tom Schaul et al. "Prioritized experience replay". In: *arXiv preprint arXiv:1511.05952* (2015).

[40] Meire Fortunato et al. "Noisy networks for exploration". In: *arXiv preprint arXiv:1706.10295* (2017).

[41] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. “Fast and accurate deep network learning by exponential linear units (elus)”. In: *arXiv preprint arXiv:1511.07289* (2015).

[42] Todd Hester et al. “Learning from Demonstrations for Real World Reinforcement Learning”. In: *arXiv preprint arXiv:1704.03732* (2017).

초록

일상 생활에서 사람들을 돕기 위해, 서비스 로봇은 센서 데이터로부터 전체적인 상황 정보를 얻기 위한 지각 능력이 필요하다. 또한, 서비스 로봇이 실제로 유용한 서비스를 제공하기 위해서는 사람들과 함께하는 환경에서 자유롭게 자율주행이 가능해야 한다. 우리는 이러한 지각능력과 자율 주행을 위한 딥 러닝 방법을 제안한다. 지각 능력을 위한 딥 러닝 프레임워크인 IPSRO (Integrated Perception for Service RObots)는 여러 가지 지각 기능을 수행하는 딥 러닝 모듈들을 통합하여 유용하고 풍부한 지각 정보들을 추출해낼 수 있다. 또한, IPSRO는 추출된 정보들을 조합하여 복잡한 지각 기능을 수행할 수 있다. 자율 주행을 위한 프레임워크인 NavNet은 딥 강화학습을 사용하여 실내 환경에서 고비용의 레이저 센서나 복잡한 mapping 및 planning 알고리즘 없이도 자율 주행이 가능하다. 우리는 두 가지 시스템의 성능을 각각 가정 환경을 모방한 실험 환경에서 테스트 한다. 또한, IPSRO 프레임워크의 성능을 검증하기 위해 가정용 로봇 대회인 RoboCup@Home 2017 Social Standard Platform League에 출전한 결과를 제시하고 분석한다.

주요어: 서비스 로봇, 딥 러닝, 강화학습, 컴퓨터 비전
학번: 2016-20100