ATAI 2021-2022

VirtualHome:
Simulating Household Activities via Programs

Aitor Malo
Alex Telleria
Jon Rubio

Contents

1. Introduction **2**

2. Data Collection **3**

2.1. Dataset Analysis 4

3. Simulator **8**

3.1. Animation 9

3.2. Preparing the scene 9

3.3. Executing a program 9

3.4. Recordings 9

4. Conclusion **10**

5. References **11**

1. Introduction

This application called “Virtual-Home” is designed to simulate activities in a house that use agents to represent possible interactions or movements. Those agents are represented as humanoid avatars, in other words, those who are going to be doing what was said before (actions).

In the application there is an environment that resembles that of a house, separated into rooms, and with all kinds of objects or actions to interact (agents) related to a house. Next, you can see in the image an example of how the house or model looks in the application:

In this application, simulations work through three components mentioned formerly: agents, environments and programs, that define how agents interact with the environment.

For the first component, agents, are represented as an object in the environment. They are moving around the environment while interacting with objects and doing different actions. There are 7 different agent types implemented and you can add multiple agents in the same scene interacting at the same time.

As for the environments, we do not have infinite scenes or to create to your liking, it is composed of 7 scenes (or apartments/homes) where in each of them you have objects and activities to execute. You can remove or modify objects at the environments in use or selected.

Objects that are inside environments are of three types: static, agents can touch but not change; interactable, agents can change their state; and grabbable, agents can pick and place.

2. Data Collection

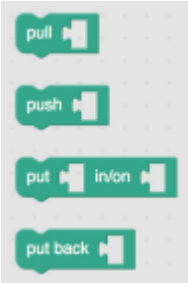
The data collection is done by crowdsourcing and is divided into two parts:

- In the first part, AMT workers are asked to give information (via verbal descriptions) about daily household activities. Each worker has to provide a common activity named with a “high level” name and describe it in detail. The activities start at a random scene from a list of 8 posible scenes:
  - Living room
  - Kitchen
  - Dining room
  - Bedroom
  - Kids bedroom
  - Bathroom
  - Entrance hall
  - Home office

- In the second part, the collected descriptions are shown to the mentioned workers and they translate the descriptions into programs. These programs will “drive” a robot that will accomplish the described activity. Lastly, more qualified workers recheck the collected data.

The programs created by the workers are composed by a sequence of steps and each step is a Scratch block from a list of 77 posible blocks. A block defines an activity and a list of arguments. For example, the block find requires an argument to specify which object is being searched for ([Find] <toothbrush>). The blocks are divided into 9 categories:

  - Communicate
  - Other
  - Body Manipulation
  - Cleaning
  - Food
  - Look
  - Electronics
  - Object Manipulation
  - Movement
  - Special Block



2.1. Dataset Analysis

Dataset has 1257 valid descriptions provided. From those, there is one program for each one corresponding to the translation of the description. However, there are 1564 additional programs corresponding to a particular set of tasks.
In total, there are 2821 resulting programs in the activity programs dataset.

However, there are also 5193 synthetic programs made by RNN, with the description provided and using only the 12 most frequent actions on the activity programs dataset.

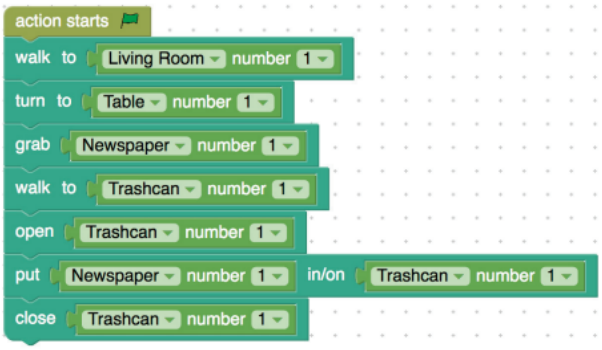
The structure of each dataset member is this way:

DESCRIPTION PROVIDED BY THE WORKER, HIGH LEVEL ACTION + DESCRIPTION

Action name:
Throw away newspaper

Description:
Take the newspaper
on the living room table
and toss it.

CORRESPONDING TRANSLATED PROGRAM, FORMED BY STEPS



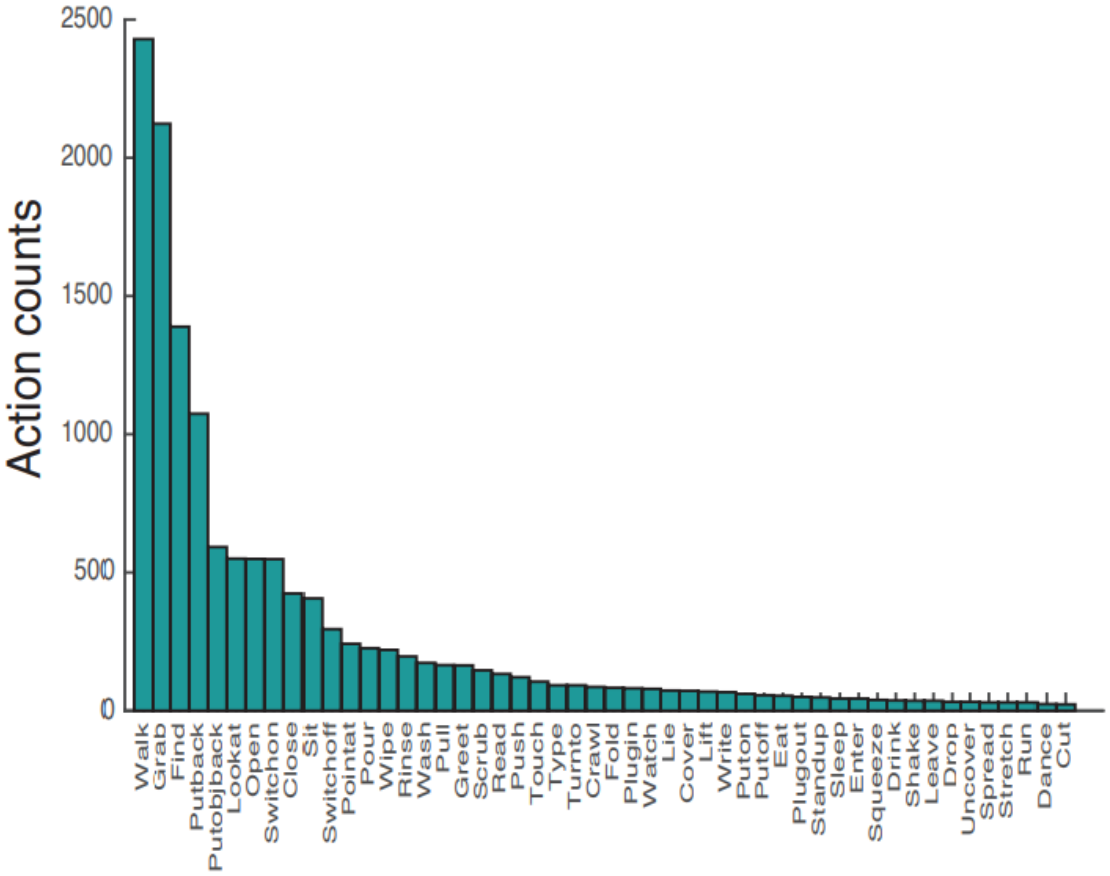
After the data collection, the activity programs dataset covers **75** atomic actions and **308** objects, making 2709 unique steps.

Atomic action: Each “high level action” is conformed by different atomic actions. For example, watching TV is a high level action. However, this action is divided by:

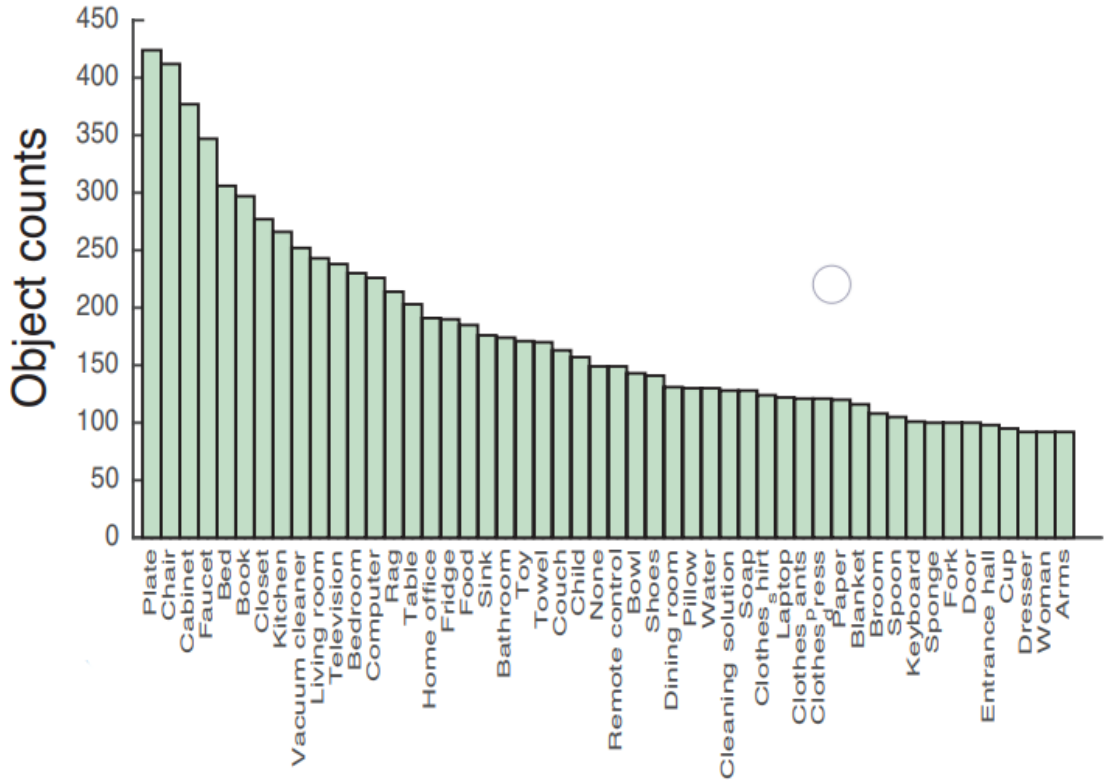
- Walk to the living room
- Find the TV remote
- Sit in the sofa
- Switch on the TV

Walk, Find, Sit and *Switch* are atomic actions.

50 most common atomic actions in the dataset



50 most common objects in the dataset



Analyzing diversity in programs referred to the same activity

The dataset contains activities with many examples, and the form to analyze their diversity is by comparing their programs. We compute their similarities as the average length of the longest common subsequences computed between all pairs of programs. We can also measure distances between activities by measuring the distance between programs. The similarity between two programs is measured as the length of their longest common subsequence of instructions divided by the length of the longest program.

Action	# Prog.	LCS	Norm. LCS
Make coffee	69	4.56	0.26
Fold laundry	11	1.29	0.08
Watch TV	128	3.65	0.40
Clean	42	0.76	0.04

Action: The activity that we are analyzing.
#Prog: Number of programs corresponding with the activity.
LCS: Average **L**ength of the longest **C**ommon **S**ubsequences computed between all pairs of programs.
Norm. LCS: LCS divided by the length of the longest program. This is normalizing.

From Videos and Descriptions to Programs (RNN)

The aim is generating a program for the activity from either a natural language description or from a video demonstration.

The model consists of an RNN encoder that encodes the input sequence into a hidden vector representation, and another RNN acting as a decoder, generating one step of the program at a time. RNN is trained with Reinforcement Learning, and is done by two steps:

1.- Firstly, we pre-train the model using cross-entropy loss at each time step of the RNN decoder.

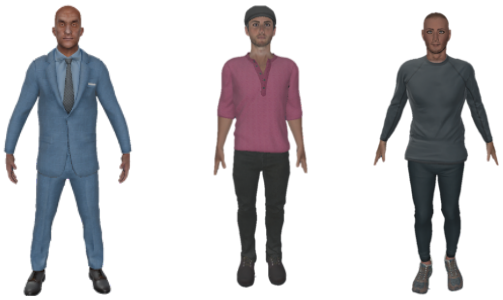
2.- Reinforcement Learning problem. To ensure that the generated program is semantically correct (follows the description/video), we use the normalized LCS metric (length of the longest common subsequence) between the two programs as our first reward). The second reward comes from our simulator, and measures whether the generated program is executable or not.

Input: Description or video.
Output: Translated program.

3. Simulator

As said before, VirtualHome is a simulator of household activities and it is made using Unity3D game engine. This engine is very interesting because it provides several useful tools such as physics and 3D models. There are 7 humanoid models (3 male, 3 female and additional invisible agent) and seven different homes.

Humanoid male models:



Humanoid female models:

Home models:

3.1. Animation

Every step in the program corresponds to an action in the virtual environment. In order to animate each action, the object involved in the action and properly animating the action are required. To obtain the correct object in each action a optimization problem is solved by taking into account all the steps and finding a feasible path. For example, if an agent has to brush his teeth, he has to grab his own toothbrush and not someone else's.

3.2. Preparing the scene

First of all, having the 3D home ready, complete the scene by placing all missing objects. To be able to complete it you have to know where to place each possible object with a knowledge base of possible locations. Where it will show us the class name and a list of other objects.

3.3. Executing a program

Before executing it, everything has to be in order, that is, the added scene, the added objects, the agents… It is necessary to have the actions of the agents with respect to the objects and the animations for each of those actions. There is a tree created for all possible interactions, in which it is used backtracking that stops as soon as a state executing the last step is found. Depending on the number of objects that are in the scene, if it is small, we can prune some interactions.

3.4. Recordings

There are between 6 and 9 cameras per room, meaning that there are approximately 26 cameras per home. To record the animation, a random camera that sees the agent is selected and it is kept until the agent is not visible. In order to record interactions between agents and objects properly, a camera with a good vision of the agent and the object is selected and its field of view is adjusted.

It is important to randomize the values of each camera so that the recordings are different from each other.

4. Conclusion

We believe that the work of this project is very interesting because through natural language processing we can get an AI to be able to distinguish and differentiate all the actions necessary to perform a certain task.

In addition, in this case, it is able to perform the actions and achieve the objective task. This can mean many facilities in other fields of technology. Simulating human behavior through these techniques can inspire larger, more ambitious projects.

5. References

[1] http://virtual-home.org/

[2] http://virtual-home.org/paper/virtualhome.pdf