Faculty of Social Sciences
Bachelor Artificial Intelligence

Academic year 2016-2017
Date: 6 May 2017

How generative models are created in Predictive Processing

Bachelor's Thesis Artificial Intelligence

Author:
M.M.C. ter Borg
3048128
Maaike.ter.Borg@student.ru.nl

Department of Artificial Intelligence
Radboud University Nijmegen

Under supervision of:
Dr. J.H.P. Kwisthout
M.E. Otworowska

Radboud University

Abstract

The predictive processing account uses generative models to get prediction errors. How these generative models are developed is still unclear. This research focuses on how generative models possibly are generated in predictive processing. We give a theory how a first very simple generative model could be generated and then test this theory using a Lego NXT robot. After this we discuss our results and conclude with recommendations to further study this theory.

Introduction

The predictive processing account in neuroscience (Clark, 2013) postulates that the brain continuously makes predictions of the inputs it will receive, based on a generative model mapping hypothesized causes to possible effects. Rather than processing all inputs bottom-up, by predicting its input it suffices to process only that part of the information that was not predicted, the so-called prediction error. The account has proven a leading theory about cortical processing, and has been successful in describing and explaining diverse phenomena and findings in neuroscience (Hohwy, 2013). Notwithstanding its empirical and theoretical successes, the Predictive Processing account is lacking one key ingredient: A coherent and consistent explanation of how the generative models that allow for making predictions about sensory input are developed in infancy.

The conceptual and computational study of predictive processing is currently a major research line within the Computational Cognitive Science group at the Donders Institute. Kwisthout, Bekkering and Van Rooij (2017) have proposed a computational framework, based on causal Bayesian networks, to capture the hierarchical, stochastic relations between hypotheses and predictions within the predictive processing account. A crucial next step in this research line is to investigate (from a computational perspective) how these generative models can be developed and refined in infancy, and what the behavioural and neuro-physiological consequences of such development would be. For example, in a current (unpublished) study Kwisthout proposes that generative models develop by becoming more detailed, predicting a specific pattern in prediction errors that loosely corresponds to the U-shape pattern of development in infants (von Hofsten, 1984).

In this bachelor thesis project I contributed to these investigations, by contributing on the question “how can individual experiences be aggregated in a crude first Bayesian network?” Since there is little done on this matter I will start in this bachelor thesis by doing a pilot study. In this study I will focus on the question: “Can individual experiences be aggregated in a crude first Bayesian network by using *k*-means?”. My hypothesis is that it is possible to aggregate individual experiences in a crude first Bayesian network by using *k*-means.

This project was part of a larger bachelor project where several students studied various aspects of predictive processing (*see appendix A: project team plan*). I used the so-called *Robo-havioral methodology* (Otworowska et al., 2015) to explore the consequences of ‘design choices’ in the theory by making it operational in a working robot and exploring the behavioural patterns of this robot.

The remainder of this thesis is structured as follows. In Section 2 I will describe a general theory of how generative models could be developed, by clustering individual experiences into a very simple causal model with one binary cause variable (=hypothesis) and one binary effect variable (= prediction), and then gradually refining this model. In Section 3 I will focus on my particular contribution in this larger research proposal and the approach I took. In Section 4 I will describe the results of these investigations, and in Section 5 I conclude with interpreting these results in the context of predictive processing and by making suggestions for future research.

Theory

The predictive processing account assumes the existence of a generative model. The predictive processing account does not define how such a model is developed. To understand how a generative model is developed one must first know what a generative model actually is. A generative model is a categorical probability distribution as proposed by Friston, Rigoli, Ognibene, Mathys, FitzGerald, & Pezzulo (2015). For example the probability distribution of tossing a fair coin is P(the coin falls on head) = 0.5 and P(the coin falls on tails) = 0.5.

Kwisthout, Bekkering and van Rooij (2017) state that in order to describe the generative models in categorical probability distributions it is important to disentangle the precision of a prediction made by these distributions into level of detail and entropy. When you decrease the level of detail of the prediction, the precision of the prediction will be increased. However the information gained from this prediction is also decreased. For example when the outcome of a football game is predicted. One can predict that one team will win or lose, or one can predict that the team will win with 2-1. The possible outcomes of the latter are much higher than the two possible outcomes win or lose. The precision of the prediction for 2-1 is lower than the prediction for winning. However the information gained by the prediction of 2-1 is higher. So what is more important? High precision of high information gain? And when will the brain choose one of the other? This question has yet to be answered.

For the making of first generative models in infants Kwisthout assumes in his grant proposal (2016) these first models are initially of low detail, and thus give little information. For example when the infant predicts (some) limb movement from an increase in (some) motor cortex activity. Such a model will have low prediction errors, however, the predictions made also give little information. With this little information it is hard for the infant to try and grasp something with its right hand, because there is no prediction made for the infants right hand. Thus the second model has to have more detail. A possible next model could be: "when there is increased activity in a particular part of the infants cortex, this will lead to an increased probability of movement of a particular limb". Initially this model will give more prediction errors, however, after time the prediction errors will decrease. Kwisthout hypothesized that the increase and decrease of prediction errors when changing to next models will look as an U-shape. The transition from a first to a second model and the possible U-shape that comes with this was the focus of another project within this bachelor thesis group.

In this paper the focus will be on how to get from individual experiences in a world one does not have knowledge of to a first generative model with low detail. Our theory is that with clustering it is possible to make a probability distribution from these experiences. In our theory, an experience consists of a cause and an effect. The causes can be classified as either 'category A' or 'category B', while the effects can be classified as either 'category C' or 'category D'. In other words, when one does A or B, either C or D happens. Assume the brain has gathered a lot of these experiences. In the context of this project we assumed (without loss of generality) that the experiences are two-dimensional an thus can be plotted in an Euclidian plane. When you plot these experiences, we hypothesize that two clouds of data points can be identified. One cloud that can be classified as A and one cloud as B. The same for the binary cause variables, one cloud classified as C and one as D. For example like figure 1.

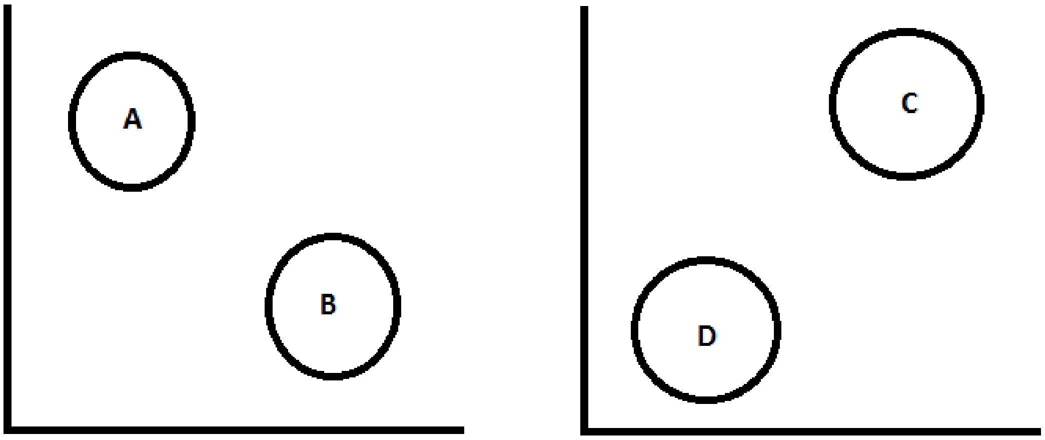


Figure 1: Expected plots of experiences.

So now there are two plots with each two clusters. These different clusters can be divided in four groups of experiences; group (A,C), (A,D), (B,C) and (B,D). Here group (A,C) represents all the experiences where one did something that was clustered in cluster A and the outcome was something that was clustered in cluster C.

From these groups one can generate a binary causal Bayesian network with two variables, a prior and a conditional probability distribution. The prior distribution represents the probability of action A and of action B. The sum of the experiences in group (A,C) and group (A,D) is the number of times one did action A. When you divide the number of experiences in group (A,C) by the number of times one did action A you get the chance C happens when one does A. The same can be applied to (A,D), or because there are only two options (A,D) also can be calculated by 1 minus (A,C). This is the conditional probability distribution over the effects of action A. For action B you can do the same calculations. These two prior and conditional probability distributions are a model of the world you're in and therefore can be seen as a very simple generative model of a part of the world.

To test the hypothesis that it is possible to aggregate individual experiences in a crude first Bayesian network by using *k*-means we will be using the robo-havorial methodology. This research methodology is proposed by Otworowska et al. (2015) as an intermediate step between theoretical and empirical investigations to reduce the gap between the two. In this methodology the theory is operationalized into a working robot to find the flaws and missing details in the theory. They state that working with robots forces ourselves to be more complete in our operationalization of the theory. Working with simulations instead of a robot also give more insight in the theory, however in simulations the real world is not included. Researchers themselves program the simulation, so it is harder to find flaws in the theory since the world is most likely not complete. Therefore we will be using a robot to test our theory and hypothesis.

Methods

As mentioned in the previous chapter I used a Lego NXT Robot to help answer the main question whether it is possible to aggregate individual experiences in a crude first Bayesian network by using the *k*-means clustering method. In the next few subsections I will describe how the robot was built, the experiments I did with this robot and how I analysed the results gathered in these experiments. Finally, I will give an overview of insights we got during the development of the robot.

Data Collection

The model

To answer my research question whether it is possible to make a first broad generative model with means of clustering by k-means I started with the conceptual idea presented in the grant proposal of Johan Kwisthout. The first model he proposed has one binary cause variable, increase in motor cortex activity, and one binary effect variable, the increase in limb movement.

For my research I also wanted to make a causal model with one binary cause variable and one binary effect variable. This model had to be possible to explore with a Lego NXT Robot. Lego NXT Robots come with multiple sensors and motors you can use to build your own robot. We tested the different sensors and found that the results with the light sensors are more reliable than with the other sensors. The distinction between dark and light was better recognized than for example the difference between a sound signal and silence. Therefore we came up with the following simple model:

An increase in movement will lead to a change in light intensity.

Building of the NXT Robot

For the robot to be able to move it was necessary that at least one motor was built into the robot. However if you only use one motor, the only thing the robot can do is move forward or backward. I wanted to have the freedom to move the robot however I wanted so I choose for 2 different motors, each attached to a wheel. This way the robot is also able to make a turn and move the wheels independently. The left motor and wheel is called Motor A, the right motor and wheel is called Motor C. There is a third wheel in the middle of the back of the robot, this wheel is built in for stability of the robot and helps the robot turn. This wheel is not attached to a motor and only moves along with the rotation of the robot.

To measure the light intensity I needed to build in a light sensor. I wanted this sensor to be on top of the robot, as if the sensor were really the robots eyes. The sensor is placed horizontally. This sensor is called Sensor 2. For comparison I added another light sensor at the bottom of the robot, facing the ground. This is sensor 1. This way I can plot my results for sensor 2 against the results of sensor 1. The final robot is shown in figure 2.

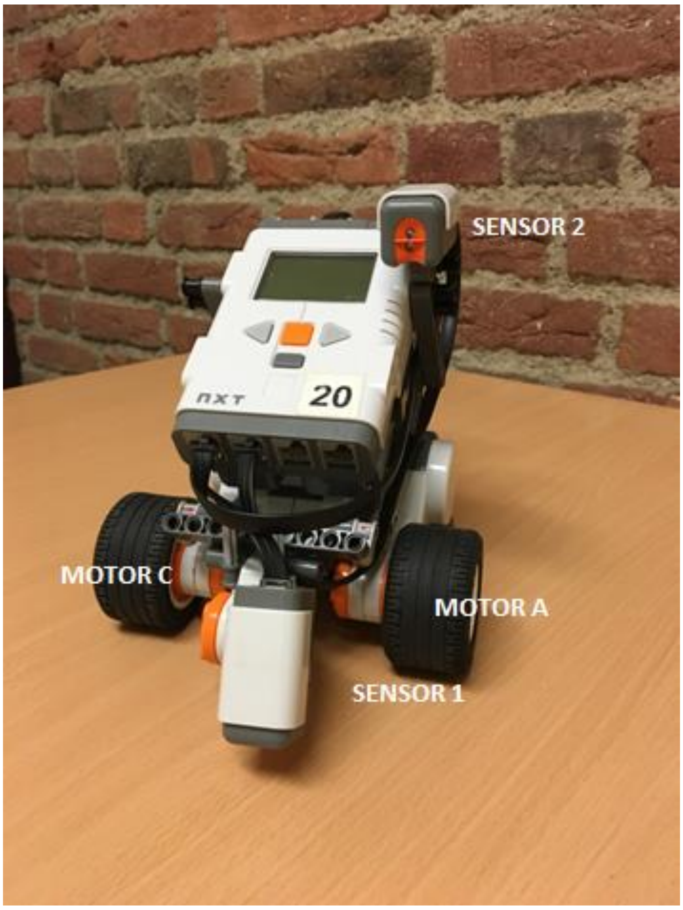


Figure 2: Final robot.

Programming Environment

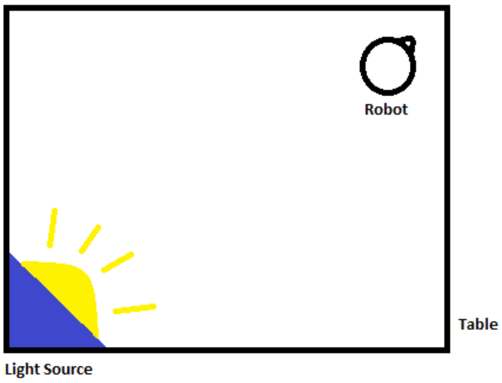
The robot comes standard with its own programming environment. In this environment, however one cannot make a connection with the Predictive Processing toolbox that was used by some members of the group. Therefore we chose to install a different operating system on the robot. We chose for Lejos, since Lejos is based on Java. All group members are sufficiently familiar with programming in Java. Lejos can be used in the integrated development Eclipse which all the group members worked in before.

Experiments

The robot had no knowledge of how the world looked and also had no initial generative models. We wanted to experiment if it was possible to create the following generative model for the robot: “If I use my actuators, the light intensity my sensor perceives will change”. Where the binary cause variable is “using of actuators” and the binary effect variable is “change of light intensity”. In this section I will describe the world the robot was in and the experiments I did.

The World

Since this is a first study, we wanted the world of the robot as simple as possible. For the change of light intensity there was only one light signal. We wanted the contrast between the light signal and the rest of the world as high as possible; therefore the rest of the world was completely dark. The light source was at one side of the world, where the robot was at the other side. One team member wanted to investigate if having a goal for the robot can further define this basic model. A possible goal for the robot could be to maximize the light intensity. Therefore the robot stood with it sensor in the opposite direction of the light source. For the movement of the robot we eventually decided on a circular movement of the robot. There are two ways for the robot to move in a circle. The first is that the robot only moves one of its motors; it does not matter in which direction. The second is that the robot moves both its motors but in opposite directions. So if Motor A has a positive rotation angle, Motor B has a negative rotation angle. I wanted to experiment both cases separately in two experiments, which I will both describe in the next two subsections. The schematic setup used in the experiments is shown in figure 3 and a photo of the setup is shown in figure 4.



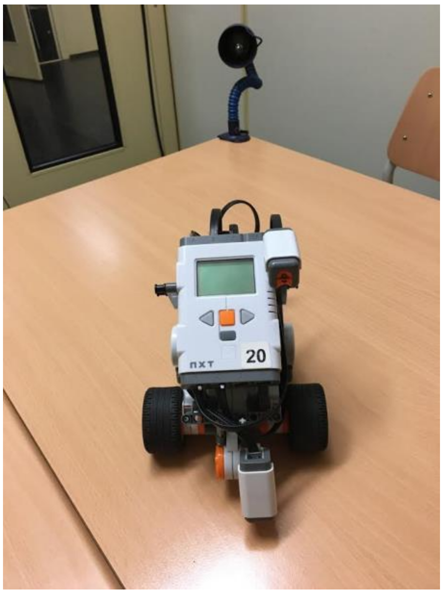


Figure 3: Schematic setup.

Figure 4: Photograph of setup.

Experiment 1

In experiment 1 the robot only used one motor. This resulted in a rotating movement of the robot. The command for turning a wheel of the robot is as follows: “Motor.A.rotate(x,true)”, where x is an angle. For the robot to make a full circle I made the following calculation: I measured the distance between the wheels as seen in figure 5.

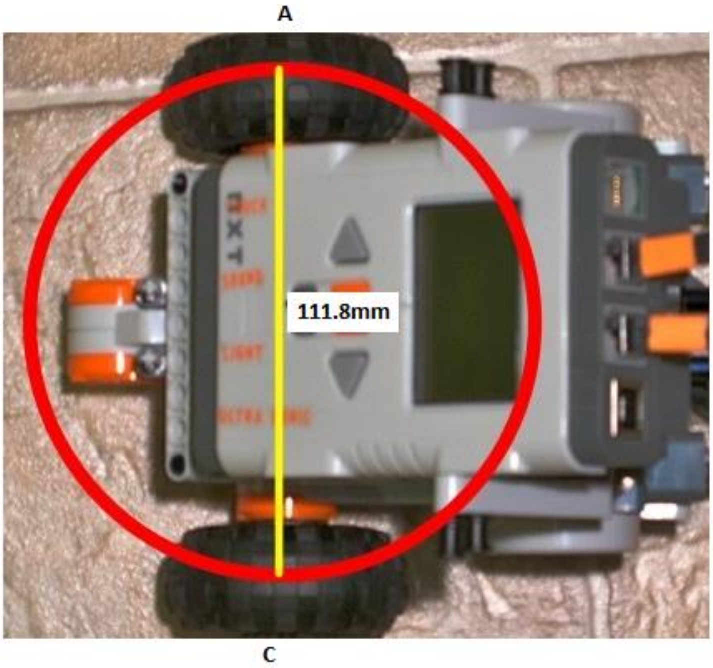


Figure 5: Measuring the distance between the wheels of the robot.

This is the radius of the circle the wheel has to travel for the robot to make a full circle. The circumference of a circle is measured by 2π*radius, so the circumference of this circle is 2π*111.8mm = 223.6π mm. The wheel itself has a diameter of 49.7mm. The circumference of the wheel is 49.7π mm. So if the wheel rotates a full circle, 360 degrees, it has travelled 49.7π mm. For a full circle of the robot the calculation is as follows:

$$\frac{223.6\pi}{49.7\pi} \times 360^\circ \approx 1620^\circ$$

So when the wheel has turned x = 1620 degrees, the robot has made a full circle.

Experiment 1 consisted of three parts. In part 1 the robot measured the light intensity at the start position of both sensors. Then the robot rotated its wheel 180 degrees and then measured the light intensity again. The robot continuously repeated this action till it reached its starting point, at 1620 degrees when a full circle was made as seen in figure 6.

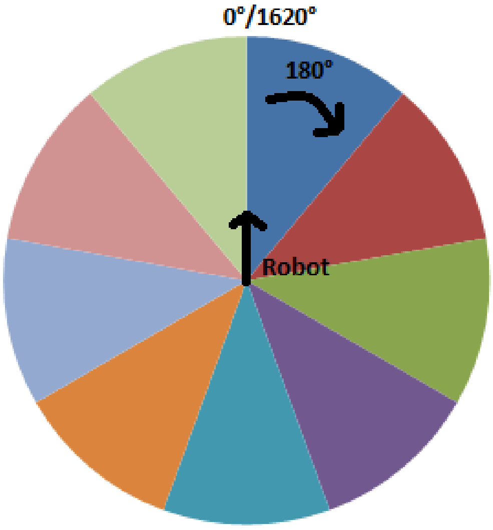


Figure 6: Circular movement of the robot divided in steps.

In part 2 the robot again started with measuring the light intensity at the start position. Then the robot rotated its wheel 180 degrees and then measured the light intensity again. The difference with part 1 is that after this first rotation the robot did not stay at this position, but returned to its start position. From there the robot rotated with its previous rotation added with 180 degrees. So its second rotation was 360 degrees. The robot did this till the rotation was 1620 degrees. The values of the light intensity in part 1 and part 2 should be the same. This part was a control experiment to check if the robot was functioning as it should be.

In part 3 the robot started at its starting position for every rotation, but instead of rotating a fixed amount of degrees it rotated its wheel with a random amount of degrees.

Experiment 2

In experiment 2 the setup of the experiment was the same as in experiment 1. Instead of rotating one wheel as in experiment 1, the robot rotated two wheels in this experiment. For every rotation the robot started at its starting position. In order for the robot to move in a circular motion, the rotation of the wheels had to be set in opposite direction. This means that when wheel A rotated with a positive amount of degrees, wheel C rotated with a negative amount of degrees. For this movement I made a new calculation for when the robot reached its starting point. In this case the distance between the wheels is not the radius of the circle, but the diameter. So instead of 223.6π mm the diameter of the circle is 111.8π mm.

$$\frac{111.8\pi}{49.7\pi} \times 360° \approx 810°$$

So when one wheel turns 810 degrees and the other one -810 degrees, the robot stands at his starting point again. This experiment consisted of one part where the robot rotated its wheels with fixed amounts of degrees and one part where the rotation was random. For the fixed part I used the same rotation steps as in experiment 1, so with steps of 180 degrees. I used every possible combination of the 2 motors with steps of 180 degrees and one extra, the start position with 810 degrees.

Analysing the data

After running the experiments values are obtained for both the sensors and the motors. Together these are the experiences the robot has of his world, the experiment setup. The goal is to make a broad generative model from these experiences of the robot; “When the robot uses his actuators, the light intensity will change”. As mentioned before there has little been done to investigate how such a model comes to existence. My research is a pilot to test if it is possible to make a first generative model from these experiences by means of clustering, specifically by using k-means. In the next subsections I will explain how I analysed the experiences of the robot.

Clustering the data

K-means

To answer my research question whether it is possible to make a model by means of *k*-means clustering it is important to know what exactly *k*-means is and how it is used. *K*-means is an algorithm that clusters data that consists of *n* data points into *k* clusters. The number of clusters, *k*, needs to be chosen before the algorithm starts. The way *k*-means assigns the data points to a cluster is by comparing the value of the data point to the centroid of the clusters and then chooses the nearest cluster. After this the means of each cluster are recalculated and the points are again compared to the means and again clustered. The algorithm will keep doing this till there is a convergence.

There are multiple ways to set the initial centroids of the clusters and to measure the distance between the data points and the centroids. For example one can choose the initial centroids of the *k* clusters random within the data domain. Or you can take a random data point from your data set as a centroid.

For this research I used MATLAB to analyse my results. MATLAB has both a pre-programmed function to use *k*-means as well as a pre-programmed function to make plots, which both were useful for my research. MATLAB uses by default *k*-means++ to set the initial centroids of the clusters and the squared Euclidean distance measure to determine the distances. *K*-means++ is proposed by Arthur and Vassilvitskii as a way to set the initial centroids of the *k*-means algorithm. They proved that their approach improved both the speed and accuracy of the *k*-means algorithm. Since the experiences of the robot are two dimensional, the Euclidean distance measure is suitable to measure the distance between the data. Therefore I will be using the standard function of MATLAB for *k*-means to cluster the experiences for the robot.

The value of *k* has to be chosen before the algorithm starts. Since the values of the experiment are binary, I want two clusters and therefore I used the *k*-means clustering with k=2.

Clustering Sensors and Motors

For the clustering of the sensor I expected that the values of sensor 1 would be constant (i.e., independent of the robot's position relative to the light source) and that the values of sensor 2 would be high when the robot faces the light source and would be low when the robot is turned away from the light source. With these values I expected that *k*-means would cluster the experiences into clusters as seen in figure 7. Here cluster 1 is denoted as 'lots of light' and cluster 2 is denoted as 'little light'.

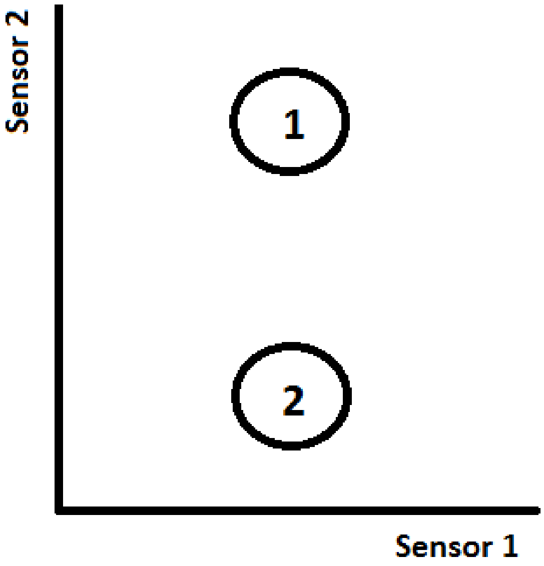


Figure 7: Expected clustering of the sensors.

Clustering Motors

For the motors we also expected to get two clusters. One cluster where the robot has little movement and one cluster where the robot has a lot of movement. For experiment 1 we expected to see the same as in figure 7, since in this experiment one motor has a constant value. In experiment 2 we expected to see something like figure 8.

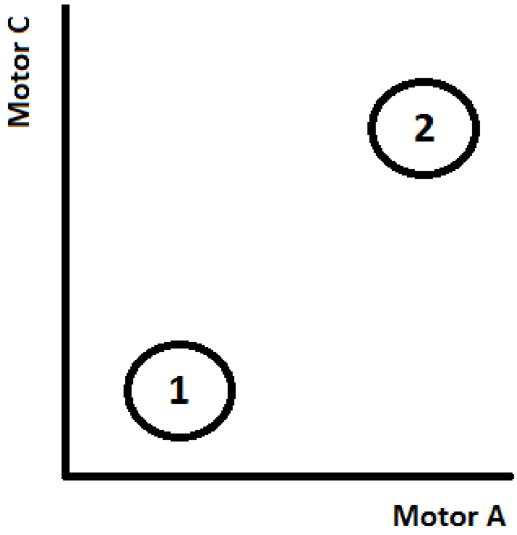


Figure 8: Expected clustering of the motors.

Insights

When I did the experiments, the clusters of the robot where not as expected. At first sight there appeared to be no clusters at all, the values were completely random. Although at first sight there were no clusters visible, I still tried to cluster them with *k*-means. The algorithm would give different answers each time it was run. When we analysed what we were trying to do, it made sense that there were no clusters, since the movements of the robot where programmed to be completely random. And therefore the values of the motors would be completely random. We think that without the use of the robo-havorial method we would not have seen this error in our thinking.

We still wanted to cluster the movements of the robot, so we could look if there could be a connection between the clusters of the sensors and the motors. For the movement to be binary there are multiple possibilities. One example; the robot moves or the robot does not move. However, this is not an evenly distributed partition. A random number is much more likely to be greater than zero than precisely zero. For the results I want the two partitions to be even likely to happen. Since the robot turns around its axis, the sensor of the robot is even likely to face each of the halves of the circle. However there are a lot of ways to divide a circle in two halves. For this world I wanted the outcomes of the values of the light source in each of the circles halves evenly distributed, as this effectively maximized the entropy of the resulting distribution; in this manner I adhered to the Jaynesian principle of maximum entropy (Jaynes, 2005). Therefore I decided on the partition as seen in figure 9. In experiment 1 this is easily implemented. Only one motor turned, so the two halves split at exactly the maximum rotation of the motor divided by two. So one half of the circle is from 0 to 810 degrees and the other half is from 810 to 1620 degrees. Only Motor A turned in this experiment. Since Motor A turned with a negative amount of degrees the absolute value of Motor A had to be taken.

In experiment 2 both the motors turned. As mentioned before when Motor A turns -810 degrees and Motor C turns 810 degrees, the robot would be at his start position. Adding the absolute values of -810 and 810 gives 1620, so when Motor A and C together turn 1620 degrees the robot is at his start position. When Motor A and Motor C together turn the half of 1620 degrees, the robot would be in the middle of the circle. So one half of the circle is when the absolute values for Motor A and C are together from 0 to 810 degrees and the other half of the circle is when the absolute values are together from 810 to 1620 degrees.

In the remaining part of this thesis I will call partition 1 of the circle seen in figure 9 little motor movement and partition 2 a lot of motor movement.

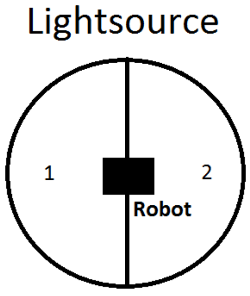


Figure 9: Partition for the clustering of the motors.

Predictive Processing

When the clusters are made, I want to look at the chances of the data points to be in a combined cluster. So for example the chance that a data point is as well in motor cluster 1 as in sensor cluster 1. I will categorize all the data points in each of the four possibilities; (1,1), (1,2), (2,1) and (2,2). This will result in two probability distributions. Where the chance on (1,1) + (1,2) equals 1 and the chance on (2,1) + (2,2) equals 1. This distributions are the belief of the robot and his first generative model. With this distribution I will calculate the size of the prediction errors with the Kullback Leibler divergence.

$$D_{\mathrm{KL}}\left(\Pr_{(\mathrm{Obs})}\|\Pr_{(\mathrm{Pred})}\right)=\sum_{\mathbf{p}\in\Omega(\mathrm{Obs})}\Pr_{\mathrm{Obs}}(\mathbf{p})\log\left(\frac{\Pr_{\mathrm{Obs}}(\mathbf{p})}{\Pr_{\mathrm{Pred}}(\mathbf{p})}\right)$$

For both a lot of movement as little movement of the robot, there are two possible observations. One where the robot observes little light and one where the robots observes lots of light. For each of the possible observations the corresponding size of the prediction error can be calculated. The actual size of the prediction error is the one that corresponds with what was actual observed.

Insights

During the setup and execution of these experiments there have been some minor setbacks in relation to the programming of the robot and the robot. I will give an overview of the insights I had during the research.

The first insight concerns the programming of the robot. While we were programming we wanted to let the robot move straight forward till a certain point was reached. So at first we used the command to start the motor and when the point was reached the command to stop the motor. We did this for both motor A and motor C. If both motors would use the same speed, this should result in a straight line. However there was a deviation in the path. We discovered that there was a slight delay between the motor commands that were given to the robot and that this results in a non-straight path. This command of the robot is also very reliable on the battery power. So when the robots battery power was high, the robot would cover a longer distance than when the robots battery power was low. Since we want a reliable outcome of the movement the robot made we looked into other ways to control the motors. We found that if we let the motors turn a certain angle this would be more reliable. However there is always friction between the surface and the wheels. Therefore it still is not as reliable as I would like it to be.

The second insight concerns the turning of the robot in experiment 2. In experiment 2 the robot will turn both its wheels. For this I gave in the section of experiment 2 a calculation of how much each wheel has to turn to make a full circle. In the clustering of the motors I mentioned that you can know which way the robot is facing by adding the absolute values of both the values of the motors. However although the outcome of which way the robot faces calculated was the same as I saw during the experiment, the exact position of the robot was not what I expected. The calculation of the circle only works when Motor A turns exactly minus the value of Motor C. If the two values differ from each other, the robot will turn in two phases. Let the value of Motor A be -180 and the value of Motor C be 230. In phase one the robot will turn as I calculated for experiment 2, with the lowest absolute value of the two motors. In this case the robot will first turn with -180 and 180 degrees. After this in phase two the robot will turn the rest of the value of motor C, 230-180 = 50 degrees. Since one of the motors is not moving, the robot will then turn as I calculated for experiment 1. For this experiment the exact position of the robot is not relevant, but for further research it is important to be aware of this when you do need the exact position.

Results

In this chapter I first will give a step by step explanation how I used the methods described in the Methods chapter to get my results for experiment 2. After this I will give the end results for experiment 1.

Experiment 2

Values Sensors and Motors

In experiment 1 the rotation of the motors of the robot are fixed in part 1 and at random generated in part 2. These values are in columns Motor A and Motor C. After this rotation the robot measures the light intensity for both its sensors, these values are in columns Sensor 1 and Sensor 2.

Part 1

	Motor A	Motor C	Sensor 1	Sensor 2
1	0	0	43	20
2	-180	180	43	24
3	-360	360	45	50
4	-540	540	43	25
5	-720	720	43	21
6	-810	810	43	20
7	-180	360	44	31
8	-180	540	44	51
9	-180	720	45	40
10	-360	180	43	35
11	-360	540	44	37
12	-360	720	43	26
13	-540	360	43	30
14	-540	180	44	48
15	-540	720	42	24
16	-720	360	43	24
17	-720	540	43	23
18	-720	180	43	31

Table 1: The results of experiment 2 part 1.

Part 2

	Motor A	Motor C	Sensor 1	Sensor 2
1	-728	238	43	29
2	-337	799	44	25
3	-268	56	44	24
4	-569	61	44	47
5	-508	470	44	29
6	-586	439	44	27
7	-118	702	45	47
8	-42	2	43	20
9	-634	875	44	28
10	-158	241	43	25
11	-616	899	44	28
12	-594	23	43	43
13	-70	773	45	49
14	-5	70	43	20
15	-402	300	44	50
16	-540	505	44	27
17	-720	164	44	31
18	-268	85	43	23
19	-772	177	44	27
20	-754	201	44	28
21	-629	787	43	20
22	-523	127	44	48
23	-580	44	44	46
24	-326	3	43	23
25	-153	653	45	49

Table 2: The results of experiment 2 part 2.

Clustering of the Sensors

	Sensor 1	Sensor 2	Cluster
1	43	29	2
2	44	25	2
3	44	24	2
4	44	47	1
5	44	29	2
6	44	27	2
7	45	47	1
8	43	20	2
9	44	28	2
10	43	25	2

Table 3: The clustering of the sensors after using MATLAB.

The clustering of the sensors is done with MATLAB. The values of part 1 and part 2 are given to MATLAB and with the *k*-means function of MATLAB you will get the following clustering, where cluster 1 is ‘lots of light’ and cluster 2 ‘little light’. All the results can be found in the appendix, for readability I only include the first 10 results in this and the next subsections. All the data points clustered and plotted can be seen in figure 10.

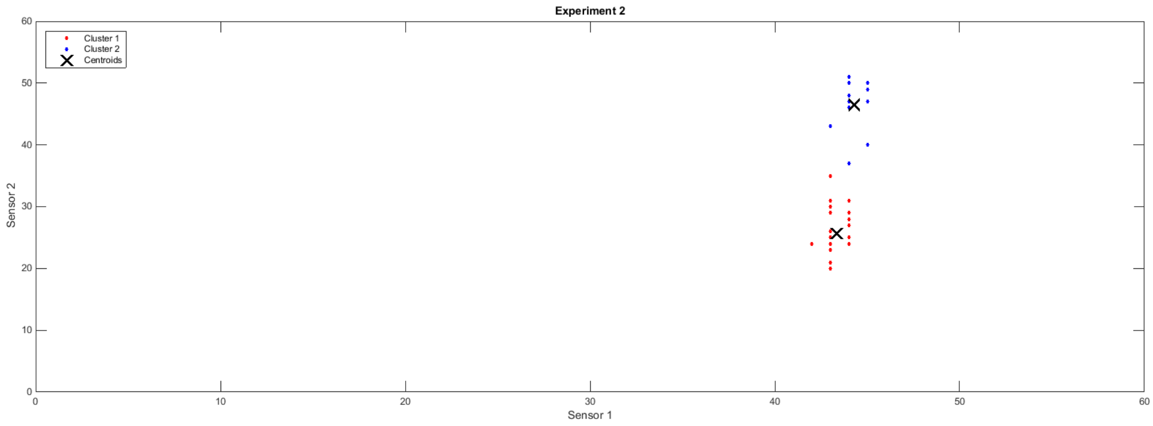


Figure 10: Plot after clustering the sensor values. Red is cluster 1 and blue is cluster 2.

Clustering of the Motors

For the Motors I want to work with positive values. Since Motor A has a negative value I first took the absolute value of Motor A. Then the absolute value of Motor A and the value of

Motor C are added. When this number is smaller than 810 the values are in cluster 1, when this number is bigger than 810 the values are in cluster 2.

	Motor A	Motor C	Abs (A)	Abs (A) + C	Cluster
1	-728	238	728	966	2
2	-337	799	337	1136	2
3	-268	56	268	324	1
4	-569	61	569	630	1
5	-508	470	508	978	2
6	-586	439	586	1025	2
7	-118	702	118	820	2
8	-42	2	42	44	1
9	-634	875	634	1509	2
10	-158	241	158	399	1

Table 3: Clustering the motor values.

Combining the Clusters

In this subsection I checked how many data points are as well in motor cluster 1 as in sensor cluster 1 etcetera. This will give table 5.

	Motor	Sensor		1,1	1,2	2,1	2,2
1	2	2		0	0	0	1
2	2	2		0	0	0	1
3	1	2		0	1	0	0
4	1	1		1	0	0	0
5	2	2		0	0	0	1
6	2	2		0	0	0	1
7	2	1		0	0	1	0
8	1	2		0	1	0	0
9	2	2		0	0	0	1
10	1	2		0	1	0	0

Table 4: Categorizing the combination of motor and sensor clusters.

In total there were 19 results for motor cluster 1, where 9 results were also in sensor cluster 1 and 10 results in sensor cluster 2. For motor cluster 2 there were 24 results, 20 of which were also in sensor cluster 2 and 4 of which were in sensor cluster 1.

Probability distribution

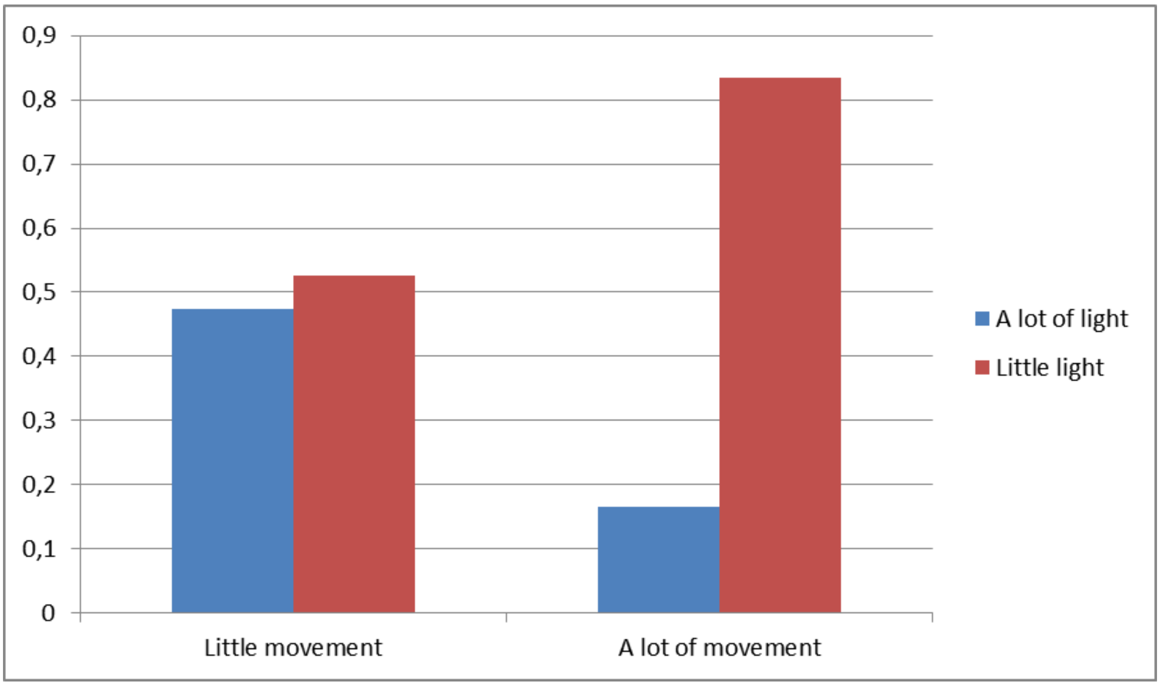
This gives the following probability distributions:

When given that the robot has little movement, what is the chance on little or lots of light.

Little movement, lots of light	Little movement, Little light	
P(1,1)	P(1,2)	P(1,1)+P(1,2) = 1
0,473684	0,526316	

When given that the robot has a lot of movement, what is the chance on little or lots of light.

A lot of movement, lots of light	A lot of movement, Little light	
P(2,1)	P(2,2)	P(2,1)+P(2,2) = 1
0,166667	0,833333	



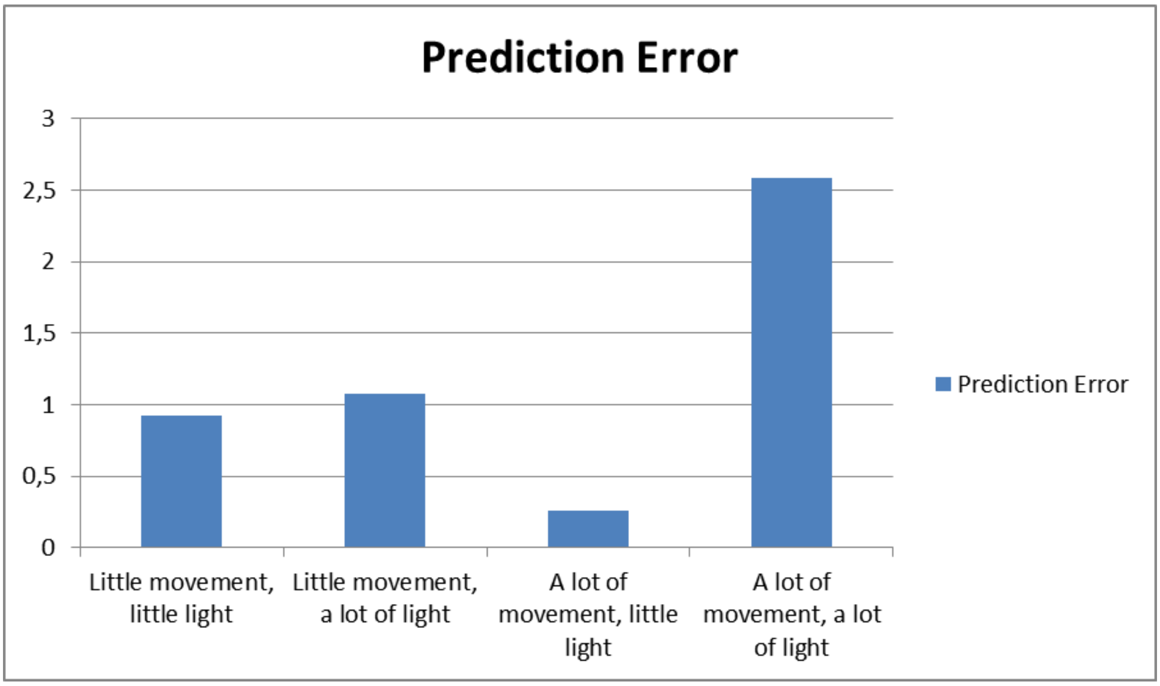
Graph 1: Probability distributions for 'Little movement' and 'a lot of movement' after experiment 2.

Size of Prediction errors

Prediction Error = Observed*LOG((Observed/Predicted);2)

P(1,1)	P(1,2)	Observed = 1,1	Observed = 1,2
0,472684	0,526316	Prediction Error: 1,078003	Prediction Error: 0,925999

P(2,1)	P(2,2)	Observed = 2,1	Observed = 2,2
0,166667	0,83333	Prediction Error: 2,584963	Prediction Error: 0,263034



Graph 2: Prediction errors for different observations after experiment 2.

Experiment 1

The values of experiment 1 can be found in the appendix. In this section I will only give the chance distributions and the size of the prediction errors corresponding with those distributions.

Probability distribution

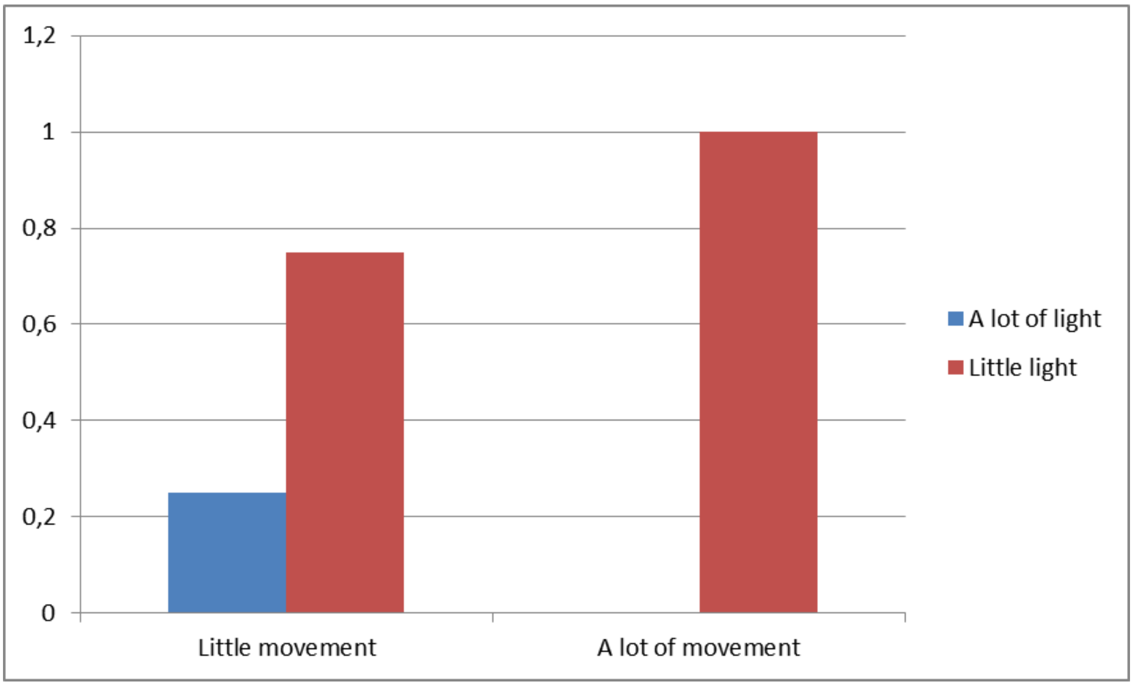
In total there were 24 results for motor cluster 1, where 6 results were also in sensor cluster 1 and 18 results in sensor cluster 2. For motor cluster 2 there were 19 results, 19 of which were also in sensor cluster 2 and 0 of which were in sensor cluster 1.

When given that the robot has Little movement, what is the chance on little or lots of light.

Little movement, lots of light	Little movement, Little light	
P(1,1)	P(1,2)	P(1,1) + P(1,2) = 1
0,25	0,75	

When given that the robot has a lot of movement, what is the chance on little or lots of light.

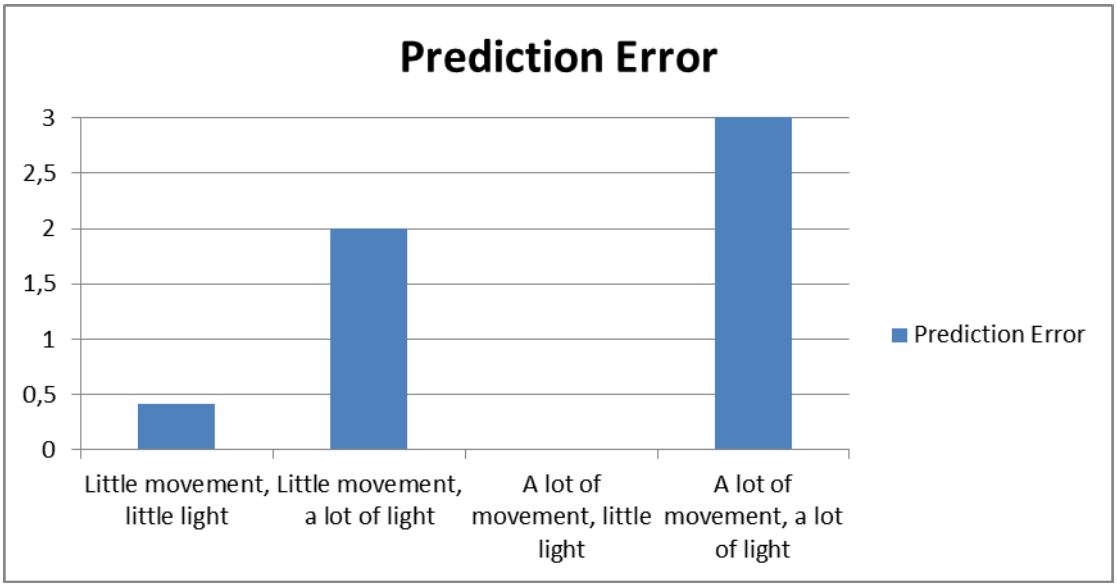
A lot of movement, lots of light	A lot of movement, Little light	
P(2,1)	P(2,2)	P(2,1) + P(2,2) = 1
0	1	



Graph 3: Probability distributions for ‘little movement’ and ‘a lot of movement’ after experiment 1.

Size of Prediction errors

P(1,1)	P(1,2)	Observed = 1,1	Observed = 1,2
0,25	0,75	Prediction Error: 2	Prediction Error: 0,415037
P(2,1)	P(2,2)	Observed = 2,1	Observed = 2,2
0	1	Prediction Error: ∞	Prediction Error: 0



Graph 4: Prediction errors for different observations after experiment 1.

Discussion and Conclusion

Discussion

The main question of this research was: “Can individual experiences be aggregated in a crude first Bayesian network by using *k*-means?” Our theory was that one can cluster the experiences of the robot by using the *k*-means clustering method with *k* = 2 and by clustering generating a prior and conditional probability distribution. We theorized that these distributions together would be a very simple first generative model. While doing the experiments we found that clustering the motor movements with *k*-means would not give clear clusters. Therefore it could be said that the answer to our research question is that it is not possible to generate a first model by only using *k*-means. However we still think that it is possible to generate this model with clustering. We discussed a different clustering method for the motor movements so we could still have clusters for both the sensors as the motors. For the clustering of the sensors the outcome of the experiment was what we expected. We saw two distinct clusters and they were placed in the plot as we expected for both experiment 1 as experiment 2.

After the clustering we generated probability distributions from these clusters for both experiment 1 as experiment 2. We expected that the outcomes for experiment 1 and experiment 2 would be very alike, even though there is another rotating motor in experiment 2. The clustering of the motor movement is based on the position of the robot, and for this clustering it does not matter how the robot will get to this position. For each of the probability distributions we expected a higher value for little light than for lots of light. In the circle of the robot there is only a small part that is lightened by the light source. This part is divided exactly in half by the partition of the circle so no matter what part of the circle the robot is in, the chance on light is even likely. Finally we expected the probability distributions for a lot of movement and little movement to be the same or almost the same.

For little movement of the robot in experiment 1 and for a lot of movement in experiment 2 the probability distributions were what we expected. The chance on little light is a lot higher than lots of light and both the distributions are somewhat alike to each other.

The other two distributions however did not look like what we expected. For experiment 1 the chance on little light was indeed higher than lots of light in the probability distribution for a lot of movement. However we expected that there would be chance on lots of light, and for this distribution the chance was zero. For experiment 2 the chance on a little light was also higher than lots of light in the probability distribution for little movement, but it was only a little bit smaller. We expected to see a higher difference.

Although the values are not exactly what we expected, we still think that the theory should not be thrown out. The probability of lots of light was very small and we only had around 40 experiences in each experiment. From these 40 experiences 20 where fixed and after re-evaluating the experiments the steps the robot did in these fixed movements where not exactly in the light source. We think that with better fixed values and with more random values the expected outcomes will be seen for all of the probability distributions.

In these experiments we used the *k*-means clustering method. For clustering the perceived light intensity we think the *k*-means clustering method showed some promising results and we think it should be further investigated in different experiment settings and with different values for *k* to see if our results also hold in these situations. For clustering the motor movements we do not recommend using the *k*-means algorithm in this particular experiment setting, since the results of *k*-means for the motor movements were not consistent. We tried a different clustering method and with this method we were able to get results. It would be interesting to compare the results of different clustering methods for the motor movements.

So to conclude: In our research it was not possible to aggregate individual experiences in a crude first Bayesian network by only using *k*-means. However we do think that a slightly altered version of our hypothesis could still hold, namely: “It is possible to aggregate individual experiences in a crude first Bayesian network by using clustering”. We think this hypothesis is worth further research. What different clustering methods could be used and what clustering method gives the best results? Also; is there a clustering method that works for both motor movement as well as perceived light intensity?

Conclusion

This was a pilot study and therefore also a really basic research. There are a lot of ways to further deepen this research. In the last subsection I already mentioned comparing different clustering methods for the experiences. Next I will describe some other possibilities for further research.

A first thing that would be interesting to look at is online *k*-means updating. In this research we had multiple experiences and after gathering the experiences we tried to generate a generative model. We used the *k*-means clustering method on all these experiences at once. Would the outcome be different if one used *k*-means after each experience?

Secondly one could simply extend the research by using more actuators and more sensors. Humans have more than two senses, so what would happen if the robot has more input from the outside world? And would this mean we need to use a different value for *k*?

The last thing I want to mention is the moving field of the robot. At our research the robot would return to its starting position after each experience and the robot only rotates in a circle around its axis. It would be interesting to see if one could also generate clusters from experiences that are gathered with random wandering around the world, without returning to its start position. In this research we clustered the motor movements by dividing the circle in two, what would be a representative clustering for motor movements if the robot does not move in a circle?

References

Arthur, D., & Vassilvitskii, S. (2007). K-means++: The advantages of careful seeding. *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms* (pp. 1027-1035). Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Clark, A. (2013). Whatever next? Predictive brains, situated agents, and the future of cognitive science. *Behavioral and Brain Sciences, 36*(3), 181-204.

Hohwy, J. (2013). *The predictive mind.* Oxford, UK: Oxford University Press.

K-means clustering and k-means++. (n.d.). Retrieved April 19, 2017, from https://nl.mathworks.com/help/stats/kmeans.html#buefthh-3

Kwisthout, J., & van Rooij, I. (2013). Predictive coding and the Bayesian brain: Intractability hurdles that are yet to overcome. Poster Presentation at CogSci 2013.

Kwisthout, J., Bekkering, H., & van Rooij, I. (2017). To be precise, the details don't matter: On predictive processing, precision, and level of detail of predictions. *Brain and Cognition*, *112*, 84-92.

Kwisthout, J., & van Rooij, I. (2015). Free energy minimization and information gain: The devil is in the details. Commentary on Friston, K., Rigoli, F., Ognibene, D., Mathys, C., FitzGerald, T., and Pezzulo, G. (2015). Active Inference and epistemic value. *Cognitive Neuroscience, 6*(4), 216-218.

Kwisthout, J. (2016). ERC Starting Grant 2016 Research proposal [Part B2]

Otworowska, M., Riemens, J., Kamphuis, C., Wolfert, P., Vuurpijl, L., & Kwisthout, J. (2015). The Robo-havioral Methodology: Developing Neuroscience Theories with FOES. *Proceedings of the 27th Benelux Conference on AI (BNAIC'15),* November 5-6, Hasselt, Belgium, Retrieved from http://www.socsci.ru.nl/johank/RoboHavioral.pdf

van Hofsten, C. (1984). Developmental changes in the organization of the prereaching movements. *Developmental Psychology, 20*(3), 378-388.

Appendix

Group Proposal

Group Project Proposal for BSc AI Thesis Research "PREDICT"

Group constellation

Supervisors: Maria Otworowska, Johan Kwisthout

Students: Ward Bannink, Maaike ter Borg, Jesse Fenneman, Sven van Herden

Project description

1. Group project title

PREDICT: Predictive Robots Empirically Determine Influential Cognitive Theories

2. Abstract: (max 100 words) *include a word count: 100*

Predictive Processing claims to be a unifying account that describes all of cognition. However, the account has been fleshed out only at the level of low-level perception. For higher cognition, e.g., action understanding, communication, and problem solving, it is sketchy at the best. In our experience, theoretical gaps, under-defined concepts, and ambiguities in a verbal theory become manifest when explicating the theory into computational models and implement them. We propose to implement and test parts of the predictive processing principle, partially using experiments with Mindstorms NXT robots. We focus on conceptual and computational aspects of model generation in the account.

3. Brief Project description: (max. 400 words)

Background and motivation

The predictive processing account has gained considerable interest in contemporary cognitive neuroscience. Its key idea is that the brain is in essence a hierarchically organized hypothesis-testing mechanism, continuously attempting to minimize the error of its predictions. More precisely, the account assumes a hierarchy of increasingly abstract (probabilistic) predictions, and the hypothesized causes that drive the predictions. At each level of the hierarchy, the predictions about the inputs are compared with the actual inputs, and possible prediction errors are minimized. A crucial aspect of the theory – how are the generative models actually developed and how are they shaped by prediction errors– is yet overlooked. In this project we focus on various aspects of this foundational issue.

Main aims and research questions of the project

We aim to study an important open problem in the predictive processing account (how are generative models developed) by means of conceptual analysis, computational and formal modeling, and robot experimentation and exploration. In addition, we investigate the application of predictive processing in the context of game AI and in the context of decision making. In particular, we study:

a) how individual experiences can merge into a general generative model;

b) how broad generative models can be further refined into more detailed models;

c) whether the behavioral pattern induced by the development of such models (in terms of in- and decrease of prediction errors and of the specificity of predictions) is consistent with findings in the developmental literature such as the U-shape of development;

d) how to integrate the concepts *exploration* and *exploitation* from the decision making literature with the predictive processing framework, in particular with sticking with a particular model versus further refining this model;

e) how different mechanisms of prediction error minimization can lead to different behavioral patterns in game AI.

Research plan (approach, methods, design, analyses)

The research approach consists of conceptual analysis, computational and formal modeling, robot-construction and exploration, and programming, in particular further development of the Predictive Processing Toolbox. **Sven** and **Maaike** will focus on research questions a, b, and c; **Jesse** on research question c and d, and **Ward** on research question e. We start the project with a replication of the pilot study in our BNAIC paper (Otworowska et al., 2015), but now with a more efficient toolbox (which has been improved in December 2015) and with “live” robot pictures. The aim of this replication is to make all group members comfortable with the toolbox and with the NXT robots.