Xzistor
LAB

Jul
2022

ANKI Vector Robot with Xzistor Concept Artificial Brain

SIMPLE DEMONSTRATION OF SUBJECTIVE EMOTIONS AND INTELLIGENCE

ROCCO VAN SCHALKWYK

1. Purpose

A short assessment was performed to establish the suitability of the ANKI Vector Robot to act as a simple robotics platform for students to investigate the concepts of agency and autonomy.

A complete, but substantially scaled down, version of the Xzistor Concept logic schema is proposed to provide the robot with simple subjective emotions and intelligence within a learning confine.

To better understand the theoretical basis behind the Xzistor Concept, the following short guides should be read ahead of proceeding with this project.

Note: A capitalised word indicates that the Xzistor Concept meaning will apply.

Preprints (free) here:

Understanding Intelligence: The simple truth behind the brain's ultimate secret

Understanding Emotions: For builders of humanoid robots

Published versions of above guides (books) available on Amazon (Kindle or paperback):

https://www.amazon.com/dp/B096QSNKQ9

https://www.amazon.com/dp/B091JPBLH1

Look also at this Preprint paper on the Xzistor Concept:

https://www.researchgate.net/publication/359271068_The_Xzistor_Concept_a_functional_brain_model_to_solve_Artificial_General_Intelligence

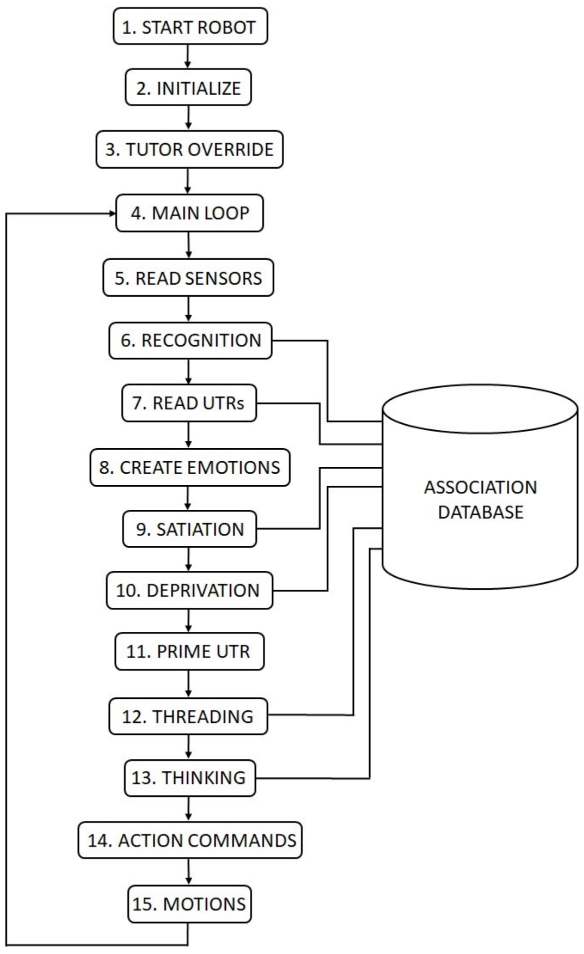
2. Vector robot system features

The Vector robotics ‘agent’ will include the robot, the PC running the Xzistor Concept brain program and the Wifi link (this link can effectively be seen as part of the nervous system).

The Vector robot instantiation will be based on the Xzistor Concept logic (simplified). This will comprise a set of sequential functional routines that, when performed at adequate pace, will emulate the biological brain. The sequence of actions is executed as a loop. In its simplest form the logic of the ‘loop’ is:

1. SENSE (the environment)
2. PLAN (calculate what actions need to be taken)
3. DO (take the actions)
4. GO BACK TO 1.

The diagram below shows a more complete description of the Xzistor logic and the functional areas addressed in this implementation:



The above logic is described I more detail in Preprint: **The Xzistor Concept: a Functional Brain Model to solve Artificial General Intelligence.**

2.1 Sensors

1. Audio sensors (4 x microphones with location detection)
2. Capacitive Tactile sensor (on back)
3. HD Camera
4. Proximity sensor
5. Edge of worktop drop-off detectors (4 in total)

2.2 Effectors

1. Front arm
2. Left wheel, right wheel
3. Face display LCD
4. Capacitive touch sensor light(s)
5. Speaker

Note: Inhibit Vector robot head from nodding to keep video camera stable for image processing.

2.3 Integrated Development Environment (SDK)

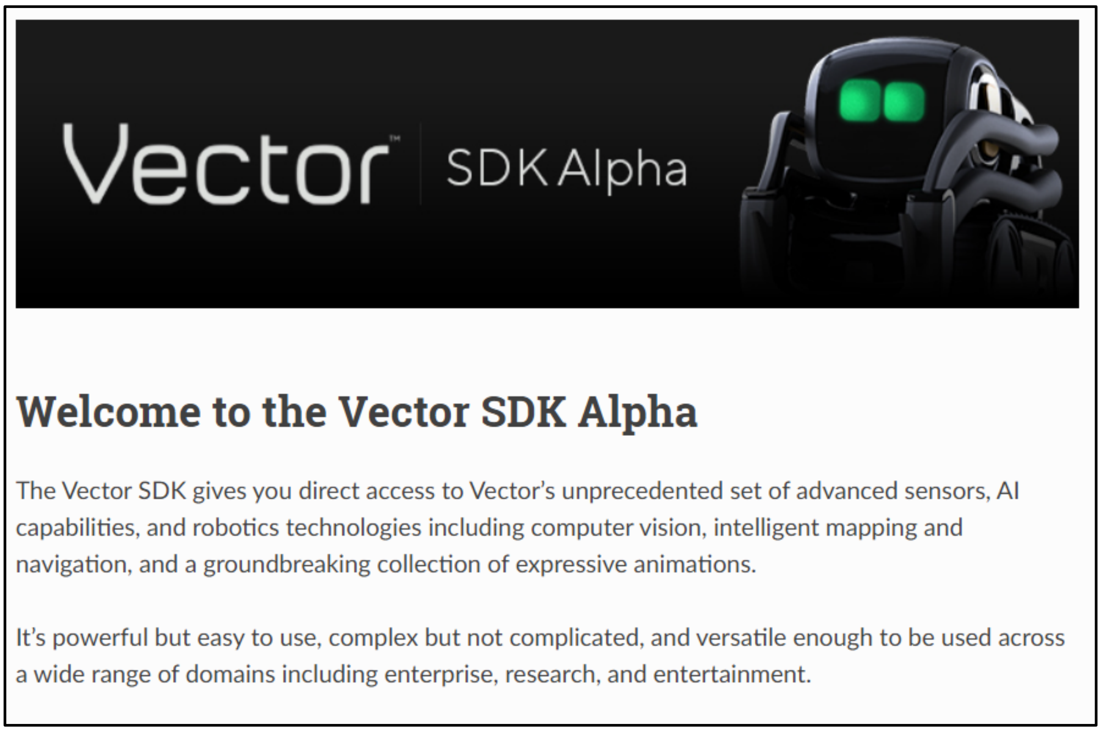
1. Eclipse IDE
2. Python
3. WiFi connection to robot
4. Many sample programs on Internet, YouTube, etc.

3. Implementing the Xzistor Concept (simplification)

3.1 Software Development Kit (SDK)

1. It is assumed that an Integrated Development Environment (IDE) like Eclipse can be used to create a Python program that will provide the Xzistor Concept functionality.
2. Sensor inputs e.g. HD Video will have to be sent to a PC via Wifi where the Python brain program will process the inputs, including simple image processing.
3. Effector outputs (e.g. left wheel and right wheel rotation speeds and direction) need to be sent to the Vector in the confine for execution.
4. The tutor must be able to 'drive' the robot over Wifi from the PC keyboard (or joystick) to allow for learning.

Link to the Vector SDK below: https://developer.anki.com/vector/docs/index.html



Welcome to the Vector SDK Alpha

The Vector SDK gives you direct access to Vector's unprecedented set of advanced sensors, AI capabilities, and robotics technologies including computer vision, intelligent mapping and navigation, and a groundbreaking collection of expressive animations.

It's powerful but easy to use, complex but not complicated, and versatile enough to be used across a wide range of domains including enterprise, research, and entertainment.

3.2 Building the Tutor Interface

1. The Tutor needs to be able to drive the Vector robot around the learning confine using differential piloting i.e. like a battle tank. Control left wheel and right wheel forward and reverse speed from the PC keyboard.

2. Use ‘keyListener class’ to interrupt and override Vector robot’s own motion commands.

3. “T”-key will mean Tutor take over, “A”-key will mean back to Vector autonomous.

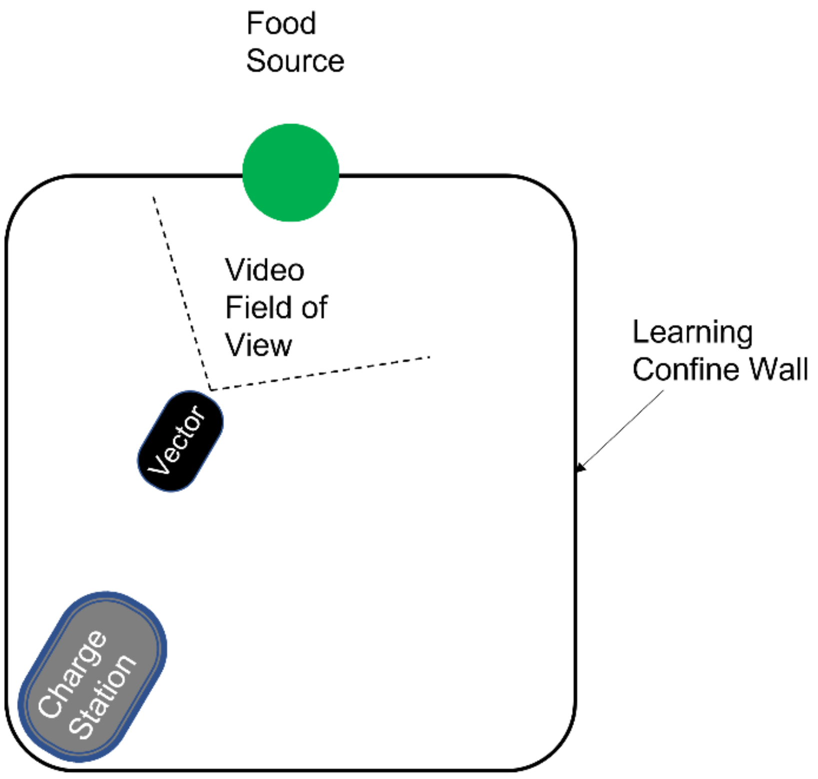
Note: Troopy has a ‘Pappa’ mode where the robot execute motion commands based on what it has learnt, and only take advice from the Tutor when it gets stuck and not sure what to do.

3.3 Building the Learning Confine (see diagram below)

1. Provide a learning confine e.g. 50cm x 50cm with 5 cm walls to safely contain the Vector robot.
2. Charging station in back corner – an instinctive charging behaviour, not part of this project. Can model as Reflex.
3. Cylindrical GREEN Food Source

Note: Another food source can be added later to see how ‘preferences’ develop.

Note: A consistent level of lighting is good to avoid confusion when Vector needs to ‘recognise’ the GREEN food source. Consider difference in video camera frame rate and light bulb frequency to avoid ‘black’ frame grabs.

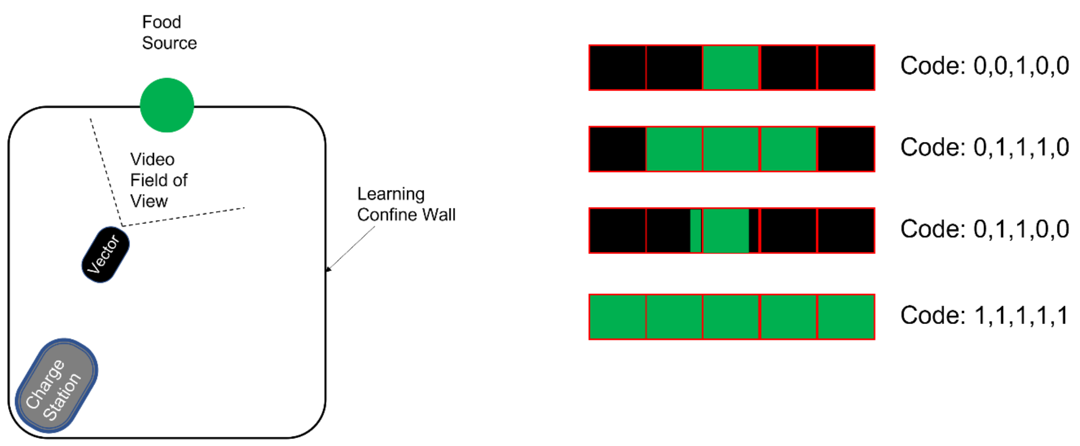


3.4 Implementing Senses

3.4.1 *Optic Sense*

1. Python code to obtain video frame grab and translate optic 5-panel mosaic sensory state to numeric representation in Xzistor brain, e.g:

For this simple implementation we can generate an optic sense that takes the video frame with every cycle and translate it into a comma separated code (representation) as shown below. The video frame is divided into 5 panels and if a panel contains green pixels, its value goes from 0 to 1. This creates a 5 digit code that the brain model uses.



3.4.2 *Touch Sense*

1. Python code to obtain distance from object e.g. wall. Distance in cm becomes numeric representation in Xzistor brain, e.g. Distance < 5cm, the Code = 1, else 0.
2. Drop sensor sensing – not used, because reserved for Emotions.
3. Capacitive touch sensor - not used, because reserved for Emotions.

3.4.3 *Sound Sense*

1. Python code to obtain loudness of sound in dB. Loudness in dB becomes numeric representation in Xzistor brain, e.g. 70 dB = Code 7. Could base shock Reflex on if sound is too loud.

Note: Hearing words and processing words can be a future enhancement.

3.5 Implement Utility Parameters

3.5.1 Hunger Utility Function (for homeostasis)

1. Hunger utility parameter: Blood carbohydrate level (can vary between 0% and 99%) variable name: int carboLevel%
2. During robot alive mode this simulated carbohydrate level depletes from 99% to 0% over 3 minutes (due to simulated metabolism). Robot will not die at 0%, just remain 99% hungry.

Note: Use 99% and not 100% so that Pain will always be stronger than Hunger.

Note: Unlike some other Xzistor implementations, we will not use the Vector robot onboard battery Level of Charge as utility parameter for Hunger to start off with. This is to keep the experiment simple.

3.5.2 Pain Utility Function (for homeostasis)

1. Pressure sensing (simulated) is used to create a tactile sensory state in the robot brain. The pressure sensing will be calculated as a function of the proximity of the Vector robot to the learning confine wall. Close proximity to the learning confine wall will also indicate extreme tactile pressure and be used as the Pain utility parameter: Wall proximity pressure (can vary between 0% and 100%) variable name: int painLevel%
2. As the Vector robot approaches the confine wall, the Pain will increase suddenly as if the proximity is akin to an impact that causes Pain e.g. proximity from wall < 5cm, painLevel% = 100%.

3.6 Implement Urgency To Restore (UTR) Functions

An Urgency To Restore (UTR) function interprets the utility parameter status and indicates to the brain how urgent it is for the brain to restore homeostasis to maintain health and survival.

Note: UTRs are not Emotion states in themselves, but Emotion states are created based on the Deprivation (DEP) and Satiation (SAT) states created by UTRs.

3.6.1 Hunger UTR Function

1. Linear function: Hunger UTR = 99% when Hunger Utility Parameter carboLevel% = 0% and Hunger UTR = 0% when Hunger Utility Parameter carboLevel% = 99%.

Note: An oesophagus delay should be built in so that the Vector robot finishes all its food before running off to Play. This means the fullness of its simulated stomach is delayed so that it stays Hungry until all its food had been consumed.

3.6.2 Pain UTR Function

1. Binary function: Pain UTR = 100% when Pain utility parameter painLevel% = 100%, else Pain UTR = 0%.

Note: In Troopy the painful collision 'lingered' because of assumed 'tissue damage' and pain decayed gradually over time. Adrenaline effects were also modelled. This can be included in the project later.

3.7 Implement Deprivation (DEP) and Satiation (SAT) States

3.7.1 Deprivation (DEP) States

1. The Hunger UTR Function is in Deprivation (DEP) when Hunger UTR Function is increasing over time. The level of Deprivation % = Urgency To Restore %, e.g. if DEP = 60% then UTR = 60%.
2. The Pain UTR Function is in Deprivation (DEP) when Pain UTR Function is increasing with time (it should be clear that the Pain UTR will shoot up very rapidly when wall impact is encountered (this is modelled), and because the level of Deprivation % = Urgency To Restore %, the follow relationship will prevail: If the DEP = 100% the UTR = 100%.

3.7.2 Satiation (SAT) States

1. It is important to remember that Satiation (SAT) is a 'rate.' The steeper the downward slope (angle) of the UTR curve, the greater the SAT value.

2. The Hunger UTR Function is in Satiation (SAT) when UTR Function is decreasing over time. The level of Satiation will be equal to d(Urgency To Restore)/dtime. So when there is no decrease in UTR% over time, the SAT% = 0. When there is an instantaneous and total decrease in UTR%, the SAT% = 100%. So SAT% values will depend on how the steep the angle of the UTR curve when being restored. For this simple application we will assume:
    a. Hunger SAT is always the same at 95% 9 (because only the GREEN food source is available, and it does not change).
    b. Pain SAT is always 90% (Satiation from Pain we tend to call 'release' rather than satisfaction or pleasure, but it remains a very positive experience as it is an improvement, and the Pain is diminished.

3.8 Implementing Reflexes

3.8.1 Deprivation Reflex States

1. The Hunger UTR Function will fire a Small Frown Reflex when Hunger DEP = 15%, fire a Big Frown Reflex when Hunger DEP = 30%, fire a Cry Reflex when Hunger DEP = 60%.
2. The Pain UTR Function will fire a verbal 'Jikes!' exclamation when Pain DEP = 100%, will make eyes BIG and face sad, and will fire a Fast Reverse for .5 seconds when Pain DEP = 100% to get away from the confine wall.

Note: We can later include Fear as the recollection from memory of a Pain learning event.

3.8.2 Satiation Reflex States

1. The Hunger UTR Function will fire a Freeze and Eat (Chew) Reflex when Hunger SAT = 95%.
2. Robot makes 'Hmmm! Hmmm!' noises while eating. Robot smiles while eating (SAT).

3.8.3 Play Reflex States

1. If the Hunger UTR of Pain UTR is below 15%, the Play Reflex will fire and make the Vector robot perform translation and rotations as if to play.
2. The net SAT effect from the Playing Reflex will be 20%. This will trigger small smile Reflex.

3.8.4 Seek Reflex States

3. If the Hunger UTR reaches 30%, the Seek Reflex will fire and make the Vector robot perform rotations in the hope that it would recognise the food source.
4. This is not strictly a Reflex but a learned behaviour, but as part of this simplified implementation it will help reduce learning time.

3.9 Implementing UTR Emotions

3.9.1 Negative UTR Emotion

The negative UTR Emotion will be a pseudo-tactile state. We call it pseudo-tactile because although it will be generated within the somatosensory functional building block (body map) area of the brain program - as if it was from a sensory input - it is in fact not based on a sensory input but rather on the Deprivation level of either the Hunger UTR or the Pain UTR. This negative Emotion state (experienced as an internal tactile sensation) is an 'avoidance' state and will become present when the robot needs to be warned to avoid something (like excessive Hunger or Pain). Because we have chosen to present the DEP status as if it is a subjective sensory state, the robot will constantly be aware of it. We say it is an avoidance state because every opportunity the robot is afforded to learn it will be mentally instructed (forced) to remember what it did to reduce or avoid this state (i.e. restore the UTR), and these actions will be recalled from memory (Association Database) and executed again in future. If the robot could speak, it would say: "I don't want to feel like this! What did I do in the past to reduce this feeling? I will try to do those same actions again right now!" This could mean avoiding impact with the confine wall or moving to the food source to eat.

It is very important to note that a negative Emotion which is generated as a subjective sensory state in this way, will be stored with the Association (memory) it is involved in during a learning event (Satiation Event). This means in future when the Associations is recalled e.g. when Vector 'recognises' an object (i.e. its tactile or optic sensory input is a close match with a tactile or optic sensory state in the Association Database) this same pseudo-tactile state will be re-evoked ('felt') just like when it was originally generated by the UTR. Originally the robot will feel this negative Emotion when e.g. physically bumping into the confine wall, but later it will feel this same negative Emotion from just 'recognising' the learning confine wall (in Vector's case the distance from the wall will generate a pressure sensation which will be recognised).

For the Vector robot the Negative Emotion will be implemented as follows:

We know the 4 drop sensors will constantly feed a drop sensor status to the sensory functional area in the Xzistor brain program. We do not need these sensors if the robot is going to remain inside its learning confine. We can instead assign them a status based on the DEP levels of the UTRs, or the Negative emotion values recalled from Associations.

We can create a pseudo-sensory state by dictating that any DEP level of the Hunger or Pain level exceeding 50%, will make all 4 sensors read 'Drop Detected!' What used to be an instant warning that a 'drop in level' has been detected, now becomes a pseudo-tactile warning to avoid whatever it is 'recognising', experiencing or doing.

Vector has a negative Emotion!

1.1.1 Positive UTR Emotion

The positive UTR Emotion will be a pseudo-tactile pursual state that will be part of the somatosensory functional area (body map), to ensure it is constantly brought to the attention of Vector and factored in when determining the next behaviour.

This state (sensation) is a 'pursual' state and will become present when the robot needs to be rewarded for doing something that reduces Hunger or Pain. Because we have chosen to present the Satiation (SAT) status as if it is a subjective sensory state, the robot will constantly be aware of it. We say it is a pursual state because every opportunity the robot is afforded to learn, it will be mentally instructed (forced) to remember what it did to achieve this state, and these actions will be recalled from memory (Association Database) and executed again in future to turn DEP into SAT. If the robot could speak, it would say: "I want to feel like this! What did I do in the past to achieve this feeling? I will try to do those same actions again right now!" This could mean navigating to the food source to eat (reward) or reversing away from the confine wall on impact (release).

For the Vector robot the Positive Emotion will be implemented as follows:

We know the capacitive touch sensor on Vector's back will constantly feed a 'touch' sensor status to the sensory functional area in the Xzistor brain program (bodymap). We do not need this sensor for our application (for now). We can instead assign this sensor a status based on the Satiation (SAT) levels of the UTRs, or the Positive Emotion values recalled from Associations.

We can create a pseudo-sensory state by dictating that any SAT level of the Hunger or Pain level exceeding 80%, will make all the capacitive touch sensor read 'Touch!'. What used to be an instant signal that a 'touch' on the back has been detected, now becomes a pseudo-tactile state and indication to pursue whatever it is 'recognising', experiencing or doing. Vector has a positive Emotion!

3.10 Implementing the Prime Emotion

There are two ways in which Emotion states can be generated, and all Emotions generated simultaneously will be synthesised by the Xzistor brain program into a net Emotional state – as simplified below:

3.10.1 Body Emotions (Original)

Utility parameters from senses and biological parameters (markers) inside the body that are part of homeostasis mechanisms will generate UTR states. Depending on whether these UTR functions are in Deprivation (DEP) or Satiation (SAT), the UTR function will signal to the brain that an 'avoidance' or 'pursual' state has been entered. The way the UTR function signals this to the brain is by setting up a pseudo-sensory tactile state that represents the Deprivation (DEP) condition or one that represents the Satiation (SAT) condition. The pseudo-sensory tactile state in the body map area of the brain that represents the Deprivation (DEP) is the negative Emotion. The pseudo-sensory tactile state in the body map area of the brain that represents the Satiation (SAT) is the positive Emotion. These Emotions are important to help the brain learn to maintain homeostasis (keep the agent happy and alive).

3.10.2 Brain Emotions (Recalled)

Because the positive or negative Emotion states are also stored as part of Associations formed while the agent learns, these emotional states can be re-evoked when the Associations are recalled. The agent will feel again the positive or negative Emotion e.g. when looking at the GREEN food source and recognising it (even when not hungry) the agent will feel the positive pseudo-tactile state being activated (akin to the pleasant body sensation a human might experience when recognising an object that has provided Satiation in the past). Recalled Emotions will therefore become another source of Emotions that will compete for the attention of the agent.

3.10.3 Determining the Prime Emotion

For the agent to know what actions to prioritise, it needs to determine its Prime Emotion. The Xzistor Concept will aggregate all the Body and Brain Emotions, and although the agent will be aware of all of these, it will select the strongest Emotion and act on it. It acts on it by going to the Association Database and recovering Associations that has in the past helped resolve or restore this Emotion. This might be a Body Emotion or a Brain Emotion. If the agent has not yet learnt to solve this Emotion, it will remain unrestored and the Emotion might become strong enough to trigger the Cry Reflex. This will encourage help from the Tutor.

3.11 Implementing Association Forming

It is easy to make the Vector robot learn inside its learning confine. It learns by storing a list of Associations - basically a set of digital states present in its mind at the time an Association is formed (normally during a Satiation Event). Associations are stored so that when they are recalled, they mentally instruct the agent to perform certain actions from its past to avoid Deprivation (DEP) and pursue Satiation (SAT).

With every program cycle, as the Vector robot sees and feels (tactile) things in its immediate surroundings, it will store Associations. Associations are therefore anchored to either an optic or tactile state, and can constitute a row of items, including the following digital states (in a comma separated array).

1. Number of Association
2. UTR Code (e.g. Hunger = 1, Pain = 2)
3. Optic State (e.g. 11001)
4. Tactile state (e.g. 2 if proximity to wall < 2cm)
5. Negative Emotion (e.g. -90%)
6. Positive Emotion (e.g. 30%)
7. Timestamp (when last recalled e.g. 37 hours ago)
8. Counter (how often recalled in total e.g. 443 times)
9. Impact Factor (= Absolute value (of Negative or Positive Emotion) x (How recent) x (How frequent)). This helps us rank Associations for relevance when constructing the 'context' around a recognised state or Thinking to solve a problem.
10. Left wheel action (motion) e.g. +30 (forward at 30 degrees per second)
11. Right wheel action (motion) e.g. -60 (reverse at 60 degrees per second)
12. Smile on face display (e.g. 50%)
13. Eyes on face display (e.g. 60% open)
14. Speaker (e.g. trigger 'Jikes!' comment = code 1)

For example, if the Vector robot is Hungry, it will refer to this Association Database, filter for the Hunger UTR, and then see if it can match the Optic State or Tactile state with what is being experienced in the environment. The aim is to 'recognise' the current sensory states in the Associations Database. If an Association can be recognised, Vector will execute the effector actions stored as part of the Association (e.g. wheel motions, speaker actions (Reflex), facial expressions (Reflex) in the hope that because it had restored the UTR in the past, it might do so again.

1.1 Implementing Threading

The Xzistor brain program will loop through its SENSE – PLAN – DO logic to solve Body Emotions and Brain Emotions. When it has no urgent Emotions to restore, the Vector robot will Play. To demonstrate the brain condition we refer to as Threading (Daydreaming), we can build in a Fatigue UTR. This will compete with Playing and make the robot tired so that it does not want to Play. It will remain idle (resting) and Daydreaming. Threading simply means running through the Association Database and recalling Associations based on a close match with the last (previous) Optic state experienced, prioritising Associations that are closely related to the last Optic State and based on a high Impact Factor. This Daydreaming will evoke Emotions but trigger now action as no Body or Brain Emotions need to be solved. Recognising an object in the environment while Daydreaming, might change the 'topic' around which the Threading takes place.

1.1 Implementing Thinking (*directed* Threading)

The Xzistor brain program will aim to recognise ques from the environment to find learned behaviours in order for the Vector robot to navigate away from the confine walls and towards the food source when it gets hungry. The moment it cannot instantly make a match around its surroundings and how it should move, it will start to Think. This means it will look at other 'close' Associations – prioritising those with high Impact Factors and close similarity with the optic and / or tactile states currently being experienced – and try the actions stored as part of these Associations. Sometimes these actions will be helpful and at other times not so helpful – this is how the Vector robot will learn. In some Xzistor applications this subset of relevant Associations are temporarily stored in a dynamic 'Context' cache and re-evoked in quick succession to help the agent 'understand' the different aspects of the problem and where it had encountered the problem before. This could provide clues to a solution. The Impact Factor of every Association will also here be used to prioritize Associations for relevance to solving the problem – these would normally have been emotionally impactful, recent and often repeated.

1.1 Implementing Effector Motions

The Xzistor brain program will aim to recognise ques from the environment to find learned behaviours (using Thinking) in order to navigate away from the learning confine walls and towards the food source when it gets hungry. If this is not required, it will Play or Rest (Daydream).

As it loops through its logic it will be driven to execute effector motions by the Associations it recalls or by virtue of Reflexes that are evoked. Reflexes will take precedence over learned Motions. When the Tutor take over control of the Vector robot, the Tutor motion commands will take precedence over both Reflexes and learned Motions.

4. Conclusion

This investigation into the viability of implementing the Xzistor Concept on a very simple robot like the ANKI Vector was a first pass assessment just to start the conversation. It indicates that in principle it should be possible to provide the Vector robot with the ability to experience subjective Emotions and Intelligence albeit at a very simplistic level. The architecture does however allow for scaling up (just more of the same) – even to the level where Machine Learning and Deep Learning functionality are integrated into areas of the embodiment. This project will require a fair-sized Python program and a full implementation (including debugging) could require extended timescales (6 months rather than 1 month). Simulations and similar robot implementations have already been achieved at the Xzistor LAB and can be drawn on. This Vector robot implementation will not just allow for an informed discussion on agency and autonomy, but also help introduce newcomers to the Xzistor Concept functional brain model and help elucidate many concepts still not fully understood by those studying the brain e.g. 'sentience', 'reasoning', 'meaning', 'consciousness' – and many more.