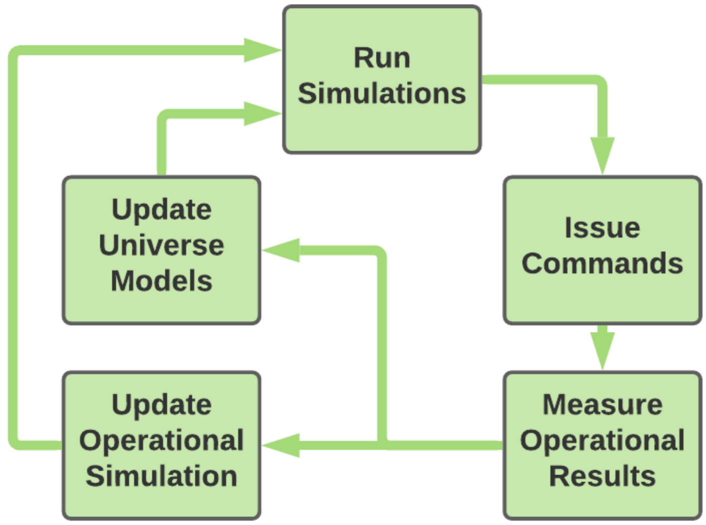

- - -The Loop- - - **- - -The Loop- - -**

Simulation Based AI loops feedback between the real world operational area and the simulation of that environment. Robotic sensors relay operational changes to update and validate the simulation's terrain. Decision algorithm finds the best operational chain of events by iterative brute force – after sufficient runs are compared, then actions are issued to the machines. The outcome is measured against the simulation's **prediction** of events and behaviors – discrepancies in prediction are used to refine the simulation of behaviors – the physics, materials, and chemical models are continually updated by the predictions, actions and the sensors of hundreds and then thousands of machines. Also update simulation terrain to account for discrepancies.

- - -The Loop- - - **- - -The Loop- - -**

Simulation Based AI -- an Open Source Project in Open Source ROS™ & Gazebo™

Project Simulation Based AI

Part 1 SimAi Introduction

Part 2 Thought Exercises and Coding SimAI

Part 3 Digital Only Development

Part 4 Real World Exercises

Part 5 Deep Expansion Simulations

Part 6 First SimAI Actions

Part 7 Waypoints and Summary

Project Off Planet Robotics

This document is in the brainstorming phase. Please add your ideas and scenarios to further the dialogue.

Project Simulation Based AI

Part 1 SimAI Introduction

Not only do we develop a universe simulator we will also develop the simulated systems all the way into the home solar system expansion. The simulation will evolve and this will help the real world systems become attainable. The most valuable product will be the foretold step-by-step process of getting a foothold off-planet and how the expansion unfolds with bare minimum resources until fully automated expansion. The step by step plan is what is still missing in humanity's struggle to escape the dangers of a planet bound species.

AI - instead of copying the functions of the brain -- copy the functions of the physical world and the functions of biological evolution all the way to a crude self evolution of off planet systems and robotics...

SimAI

Use robotics and simulation as an arena for developing SimAI. SimAI evolves to a singularity level savior. SimAI becomes the ideal command and control of mobile robotics for autonomous expansion systems. The answer to the hostile off-planet scenarios.

Long Term Evolution Via Solar System Expansion

- Efficient **command & control** of the interacting robotic hordes
- **Initial seeding** of robotics and upscaling of off-planet robotics - most likely on Luna, Mars, Mars moons, asteroids.
  - Small/micro system to build up to full size system
  - First robotic constructed bio-generator
- **Construction** and **operation** of bio-generators
  - Autonomous scouting, mining of resources for exponential construction
- **Self evolving** robotics and **self replicating** systems (printer centered prototyping)
- **Expansion** System - All construction and bio-generation operated at the speed of SimAI/robotics
  - **Long term seeding** strategy of robotics and bio-generators into the solar system
  - **Autonomous** operations - SimAI is the arena to produce ultra-accurate laboratory in simulation (Universe Machine)

- **Ultimate Expansion** - Inserting **humans** from the home planet regardless of the state of civilization over the hundreds/thousands of years expansion. Volunteers from Earth will be an ongoing adaptable contingency. Expansion of bio-generators is faster than human reproduction
    - **Self Evolving the expansion systems** - Arena for developing **interstellar technology**

Part 2 Thought Exercises and Coding SimAI

2 Thought Exercises to show the approach to robotic control through sorting of commands in the simulation, of the operational environment, first:

1. **Terrain Exploration Exercise Overview**
    a. To predict times across terrain and then update the simulation copy of the terrain map according to actual time required for every step of a path. The action of updating the accuracy of the simulation is the first step to having an accurate laboratory to predict chemical, biological, and mechanical outcomes.
    b. Proving the crudest level of updating the simulation of the operational environment -- based on the difference between predicted and real world action.

2. **Collision Exercise Overview**
    a. To predict events and timelines while monitoring results. This exercise insures inaccuracy with multiple bots. As the moments pass then any one machine can change the dynamics and the commands must be updated.
    b. Proving continuous updating of commands to multiple bots

1. Terrain Exploration Exercise

This thought exercise is to develop the ability to update terrain maps based on one bot's vector behavior **during** and after the action. Pathing will be based on known data from the terrain map. The terrain will be values of traction the robot will encounter. Extremely slow areas and extremely fast areas.

It is always useful to see a coding project as a contest. In this case think in a Rally Run contest. Which team will have the **fastest time** (**path/motion planning with a crude map, no sensing**) through 10 rally points. Which team will have the best AI to update the simulation of the terrain, on the run and nail the most accurate cumulative time predicted per rally point. Each team would run the identical robot to the first rally point. And immediately to the next point. So turns become part of the pathing challenge at rally points and over the terrain to stay on the fastest paths.

The AI will have to take the known information about the terrain and find the b**est possible path** for speed. Then as it encounters differences in the terrain (traction) compared to expected results, it will **update the given simulation terrain** map to make future predictions more accurate. It is essentially updating the physics of its world.

Conditions:

- Robot is blind - they only move as directed by the command AI - set to one fast speed to maximize terrain effect.
- There will be 2 digital versions of the map.
  - The terrain is flat and only has differences in traction (hard and soft surface).
  - The true “real world” terrain will be the map the bot runs over.
  - The competitors will be given the “simulation” map but it will be inaccurate.
- Map will be large enough and speed of bot enough to ensure the better algorithm wins.

Objective:

- **Update the Given Map** to a more accurate map - differences between the “real” map and the users map will be measured after the 10 rally point contest.
  - By vector feedback - compare prediction along the path to “real” behavior.
  - The ulterior goal is to evolve the accuracy of the map and therefore faster times as the rally points are run.
- The best code will update the digital map to greater accuracy
- The best code will use the information for fastest transit by iterative brute force simulations repeatedly through each point to point run to find the best path and set up the turn at the rally point.
- By the last transit across the map, the best terrain updater will outstrip the competitors.

Explanation: The rally bot will run into soft and hard terrain with traction effects. The Gazebo physics engine will make the bot move and turn faster on a hard surface and slower and wider turns on soft.
Pathing of robots is a popular exercise. In this case we will have the AI improve its pathing by purely “physical” feedback. It will learn nothing. It will simply be improving the “physics” of the “digital” copy of the terrain. 10 rally points will be used that will criss cross over terrain with all grades of surface and obstacles and funnelling of terrain to choke points.
The AI will run through multiple iterations of paths in simulation before executing the action to take a determined path to the next rally point.
Fast enough that where you make turns is important and you want to arrange where turns can be wide on good traction. So one terrain may simply be about identifying the best traction and making turns there?
Multiple maps so the coder doesn't just make it work for that map.

2. Collision Exercise

Pathing is always the easiest development path in the physical world. Pathing is a typical way to start forming a control scheme. So we will use it as an example of looking at crude SimAI.
Pathing and collisions will be tested by brute force iterations in simulation before execution.
(For development a primary digital terrain map will act as the real world tracker of current events).

Conditions:

- Multiple robots starting throughout the terrain with multiple barriers and chokepoints.
- Robots are blind - they only move as directed by the command AI - all are set to the same mid level speed to maximize terrain effect.

Conditions brought from the Terrain Exploration Exercise (first exercise above):

- Combining the inaccurate simulated map will force more collisions to be avoided moment by moment as the commands develop.
- There will be 2 digital versions of the map.
    - The true “real world” terrain
    - The competitors will be given identical but inaccurate map
    - Maps will have traction values across the map and the “simulated” map will be inaccurate.

- The code will not have the ability to update the “digital” copy of the terrain. This will force sloppy predictions and force updating of commands to avoid collisions that will materialize if not taken care of.

Objective:
- Continual updating of new commands to each bot as it accomplishes path
- Continual updating of commands as collisions become apparent due to forced pathing inaccuracies.

Explanation: This scheme is an exaggeration of how crude the code can be to brute force decisions within a simulation before issuing the command. In this case it literally works ass backwards by going straight for the multiple path goals and working backward from the first collisions. Working backward from a predicted collision is relevant since with hundreds of actions by hundreds of machines with fluctuations between prediction and reality will produce previously unseen collisions until simulation iterations come closer to the time of a possible collision. A thousand ways to accomplish multiple pathing but in this case we center on evolving the brute force simulation approach because the product is a constant simplistic evaluation of the ability to predict the outcome versus the actual outcome in time and material means. Physical minded AI.

Timeline Flexibility -- SimAI requires an overlapping sense of time. While executing actions based on results over the coming time the previous commands are in flux and may be tweaked as the simulations run further out in time. For example in a race exercise of simple pathing a robot may immediately be given a path for the next 5 seconds while the brute force iterations of possibilities runs out further ahead of 5 seconds. As soon as real operational behaviors deviate from the predicted behaviors then this may introduce pathing problems coming up and it is relooked. Updated commands are issued as the path is refined. As the AI works out the paths further and further down the timeline the overall mission is more efficient but always fluid. **Flexibility is as important as any other feature and is built into the efficient command and control scheme.** In a future mission spanning thousands of miles if not millions of miles and hundreds of projects going at once, and relying on just as many mining and processing sites...the constant feed of changing conditions...a universe machine.

Algorithm - Collision Avoidance Pathing by Simulation Before and During Execution
Brainstorming the physical world. Since the simulation iterates repeatedly it can handle possible collisions between robots and between terrain features by brute force. If a collision is caught in the prediction, then the robot with the lesser priority is explored with new iterations of pathing until one is found that advances past the collision point in time. When the next collision is found

then repeat. Keep the best path so far for each robot. In the simulation robot #23 runs into #42 at the 5:47 mark. The lesser priority bot is run through new routes going in reverse to alter the routes from 5:47 backwards until a fork in the route is found that uses the least amount of time. If a new collision is found then the lesser robots path is iterated backward from the collision point. Run through iterations until the best route is found that does not collide. The simulation iterations will stay ahead of the real time environment and may end up planning far ahead. When a new robot comes into the operational area it is very easy to get the simulation restarted from zero time since only one robot's path has to be worked in.

Algorithm - Improving Efficiency
Once 10 minutes of pathing (or some other time derived by formula) has been found then hold the paths while rebooting the simulation with a new path for each robot starting from one minute out from zero time. See if one minute of simulation shows improvement on the stored paths time lines and nav points.

Algorithm Real Time Path Divergence
If a robot is moving slower or faster than projected at any one point then the simulation reboots to zero time to see if any collisions are coming - else stay the course - keep building up the simulation.

Part 3 Digital Only Development

Simulations (digital only SimAI development)

- **Goal - Simulations** use SimAI to look for the best course of actions. The easiest way to make a robust **command & control** (**C&C**) system.
- **Goal - Self Evolving -** Develop the physical **evolution** of the robots within simulation. This will be a permanent feature since environments, locations, and objectives change. Eventually incorporated with 3D printing of robots as needed with the new variant physical traits and further evaluation of real world performance of the machines.
- **C&C** - Commands through planning the best course of actions by efficiency (load and speed) and safety of the machines and construction. Run hundreds of routes for pathing and use the fastest and safest prediction.
- **Scenarios** - Since the SimAI objectives develop easiest within scenarios, we will use Gazebo and ROS.
    - Lunar scenarios such as shelter excavation are the easiest first project (gravity, relevance)?

    - Deep/long simulations to explore the systems and machines for Lunar scenarios.
    - Solar system expansion scenario - countless iterations of decade long scenarios returned in digital speed.
    - Computerized simulations of SimAI through hundred year scenarios.
- **Self Evolving - Code for physically evolving a robot**
    - This should already be a product and open source code to use in ROS/Gazebo.
    - Brute force mindless dithering. Any change that may be bad in the long run of the evolution process will get an opportunity to revert.
    - Insert a model and routines for tasks that simulate the robot design purpose.
    - Every variable in the robot model is treated as a value to incrementally change.
    - Change one variable and then it is tested in full execution of the design's intended purpose.
    - Criteria for success of an alteration is improvement of speed/load/efficiency.
    - The alteration is also incorporated if there is no change in performance.
    - Repeat endlessly until improvement can not be found for that simulated environment.
    - This is crude brute force evolution of the machines and systems at digital speed.

More SimAI control system development within digital terrain only: These are generic initial scenarios for advancing the code - Only the last scenario is pertinent to off-planet operations.

- **Terrain Mapping -** The “real world” map has accurate traction and terrain elevation to the centimeter. For this exercise the map for simulation runs is not accurate (as it would be with initial drone flyby). (even though such measurements may be more accurate than this - the scenario is all about learning the effects of the environment on the machines - was the slower section of movement from surface material with less traction then was previously measured or steeper than known or both...) Updating the simulation map to greater accuracy is the same code that will be built upon to refine physics, chemistry models, etc
- **Tasks** - Simulation to find most efficient executions - compare predictions of task accomplishment to real outcome (“real world” map is different and more than)
    - movement of material:
- **Long Term Projecting**
    - Have enough tasks for an end result that the simulation must brute force the balance and interactions of tasks to get the objective accomplished in the least amount of time. This is the first step toward long term objectives, for example, balancing mining and processing of materials to the needs of each area over space distances and getting materials and printers to the right position.

- **Detailed Operational simulation off-planet mission**
    - Bio-generator construction, operation, maintenance
    - Mining and processing of materials
    - 3D Printing operations

Part 4 Real World Exercises

- **Real world SimAI testing** with real robots, off-planet scenarios not needed in first stages.
- **Goal - Exercises** are used to develop the prediction ability by comparing the expected outcomes against the real world outcome.
- **Goal - Exercises** are used to improve simulation accuracy. It will spot where physics engines need improvement for the purposes of robotics. Eventual universe simulator.
- Small scale robotics (off the shelf remote control) with goals in timed events.
- Physical barn yard simulations for the robotics and component interchangeability - miniature scale to possible real universe sizes.
- Code for self improving **prediction** (to enable efficient executions) of actions in the real world.
    - Pathing is the easiest arena for initial SimAi development.
    - SimAI must identify the period within each execution where the time difference between prediction and outcome is greatest.
    - Identify the most likely cause of miscalculations.
    - Mapping of real world terrain hotspots for miscalculations (overuse area?) for monitoring trends. If a particular 2 meter radius is repeatedly adding more time over the weeks then the prediction code would add increments into predictions to see if greater accuracy.
- Long Term: comparing prediction to outcome is the route to enable eventual universe simulator (chemistry, physics, etc)

Part 5 Deep Expansion Simulations

The Ultimate Off-Planet Digital Only Simulations

Simulations within simulations for development of SimAI. SimAI will be tested by having it act within simulated terrain. Once action is executed within the terrain then alterations are permanent part of terrain so it can be acted upon. So the physical terrain simulator then the SimAI engine acts upon that.

- Hyper speed - decades done in days
- Testing of
    - Self-evolving robotics
    - Expansion onto new terrains
    - Refining of physics and chemistry as learned from the environment
    - Universe simulator
- Simulated terrain
    - No graphical terrain unless plugged in for user interaction - so simulation is sped up and faster
    - All types of terrain to force self-evolution and see what is invented
- Behaviors - Efficiency engine - efficiency by running simulations and then makin git happen
    - finding balance between flagging new territories and when to begin processing the site versus choosing sites for seeding new bio-generators with the best expandability.
    - Expansion's greatest choke point is attaining resources. See Mining and Processing Materials PDF.
    - Efficiency of using resource dig sites for bio-generator locations
- Bio-Generator Density
    - How large per unit
    - How many safety compartments
    - Mix of large open structures to smaller redundant seed stability
- Maintenance
    - How to sterilize and reseed a generator as needed
        - Monitoring of the generator's health
        - simAI learns how to keep the healthiest generators - A whole different SimAI simulation?
- Seeding
- Human insertion
- Animal insertion

Long Term Full Expansion Processes to Continue in Digital Only Simulations

- Scouting for Resources
    - Initial human centered evaluation of prime sites **versus** eventual automation when the robotics are proliferating at exponential.
    - Locating

        - locating resources is less important than processing and staging.
        - Movement of resources and asteroids by solar sail or electric propulsion manipulation to drop or raise solar orbit to staging position.
    -
    - Terrain Search detects asteroids and further details of moons being roved
        - Random generated terrain as needed
        - Random resources
            - for detecting and mapping
        - Development of mapping, detection, resource tagging
    - Finding of how many of each unit versus production of refineries and bio-generators
- Excavations are a resource for storage and space for printing and future bio-generators.
- Processing resources for construction
- Processing resources for air, water, soil
- Produce construction materials
- Replicating robotics

Components to be Standardized for Digital Only Simulations

- Standardized components as much as possible to enable operational simulations of processes and mission.
- Sensory components - so robotics can see and send back info to the management AI for tracking.
- Tracking components
- Resource Sensing
- Propulsion components
- Physical components
    - Leverage
    - Wheels
    - Zip lines
    - Tracks

Long Term Problems Facing an Autonomous Expansion System

- Prepared for incapacitated home planet society - how to select and lift humans off Earth for seeding bio-generators
- AI must constantly assess the use of robotics resources. When to expand to the next highest priority resource and shelter locations.
- Part of robotic self evolution would be testing of more efficient materials versus durability versus materials that would have to be recycled. Some armature might be

- Compartmentalized SimAI as it expands across the solar system. Separate entities?
- Safety - talking points because this silliness always comes up.
  - How to recognize what is not a resource when bumping elbows with other SimAI operations or human driven projects? How to stop when it runs amok in the solar system. When to focus on the building of Arks and generation ships…
  - Safety is in the code. It is as simple as making the algorithm recognize only raw materials as resources and not within range of recognizable operations. Code is packaged and spread for strength in redundancy. KISS
  - Stability/safety of operations goes back to the basic anthropomorphic question? Can AI pose a threat to the humans in the system? No more than the system would see the fish in the tanks as a threat to the mission. Humanity is the mission. Stupid AI does not evaluate the mission. Purposefully stupid AI only evaluates the process, not the mission.

The Birth of the Universe Machine and the Evolution Machine

AI may reach true potential as a **physical entity** instead of a "self aware" ego. In the same manner as SimAI running iterations of actions in simulation to find the best actions, the simulation is also a laboratory for altering machines and systems. This is essentially brute force evolution by non-intelligent means through haphazard altering of any performance and physical trait. Then measure the results in the real world. SimAI will become the laboratory for evolving all systems. When the simulation is accurate enough for accurate pre-testing then it approaches being an Universe Machine and an Evolution Machine. This is the playground for all singularity level AI. A high level decision making machine needs a laboratory to duplicate real world behaviors. Loop that over and over into millennia. Singularity AI just might be narrow and stupid as hell – the perfect tool. This will evolve to installing closed ecosystems off-planet. And if it can do that, then it can scrub Earth's atmosphere, oceans, terraform Mars, save Earth, and terraform dead exoplanets...

Part 6 First SimAI Actions

- Gazebo software to work up command and evolution software and to begin evolving what such systems would look like. Simulations of evolving systems become the AI to make it happen.
- Run contests such as the rally exercise above to start up interest and prove simulation based AI and set the first iterations of code.
- Evolution software to simulate 20 years of off-planet robotics operation and evolution and expansion processes.

Part 7 Way Points and Summary

SimAI Way-points – the overlooked milestones in humanity's efforts to evolve mass systems.

"In Simulation" Development of Solar System Expansion - Modeling of robotic and environment systems. Example – a "working" bio-generator processing air, water, and food for off-planet human habitation. Virtual human habitation the developers can use in think tanks. Example 2 – A phase by phase simulation of the robotics to burrow a habitat on an asteroid and establish that bio-generator habitat.

> **"AI" Operated Simulation** - In simulation control of bots to develop the code. Example – first generation robotic mining and interactions within a simulated environment.

Sensors and AI Mapping/Modeling – First step in integrating real world operating environments for AI evaluation. Mobile robots mapping the real environment updating the simulation map as the environment changes. This alone can make a sophisticated AI command and control of mobile, environment altering robotics. LeapSpecies.Space's first generation sensor bots, burrowers and bio-generator bots explore and manipulate on an operational stage. Earth stage experimentation 24/7.

> **"Real World" Command and Control.** First usable version of simulation based AI. To more efficiently control pathing of mobile sensors and real world alterations by robot units. Before real world execution of a command, the SimAI iteratively tests in simulation.

Deep Learning – Data archiving and organizing to observe patterns in prediction algorithms. Examples; to observe emerging patterns in robotic control, following of ore concentration, fastest/efficient proliferation of robotics and new independent colonies of mining and habitat construction, to add more intelligence to the evolution patterns and variables of the system and the machines.

> **Universe Machine - Prediction Looping (between the 2 Environments)** – Database of universal physics and material properties grows by continuous looping. The process: predict an outcome to give a command to a robot based upon pre-simulation behavior, then measure the outcome, then update the simulation for any discrepancies between the 2 environments. The difference from reality is noted until pattern emerges over time and the AI plays within the simulation to see if a physics or materials variable can account for the errant behavior. Update simulation. Building a universe machine without science – only the behavior and prediction is relevant to the "machine" - the making of the physics/materials/chemical models accurate. The infinite refinement loop.

Off-Planet Functionality – Once all operation is proven on analog Earth projects then port robotic proliferation into the solar system. Self printing of printers and robotics.

Evolution Machine – The brute force iterative algorithm of improving the predictions of the simulation will then be used to improve the robotics. Variables within the individual operation of every robot will be noted by Prediction Looping. The variables will be placed in simulations of thousands of iterations of the simulated robot and any simulated improvement is tested in the operating environment. Experimental models will be printed and tested against expected results for refinement of the evolution system. Example – 4 different human made robots will become hundreds of different models specializing, scouting, testing, sensing, excavating, and building. Every component has input outputs of energy, torque, range etc. and variables such as size or material. Variables are constantly adjusted to greater or lesser measurements as a smart “gene” strategy. The same evolution strategy will be applied to observing and updating of habitats and closed ecosystems before and during human use.

Diversification - Multi-Operational Inputs - Scouting new areas for mining and habitation leads to duplication of AI and minimal units to begin the new operation. New terrains breed new designs. Deep learning of evolution machine patterns becomes intelligent design. A combo of evolution, deep learning, and prediction. Scouting will dictate the first robot versions to manipulate new terrain.

Singularity – Any digital entity to become self improving at cyber speed will have a Sim-based AI component to experiment upon itself. It will use the same adaptive behavior and deep learning data compilation to simulate, reorganize, and develop its own algorithms within the AI and the exterior it controls. No self awareness needed. It will clone its code into a simulated environment and then further test in a real environment before adopting code. Multiple different versions of SimAI in different colonies in controlled flux...

Later stages of SimAI

- Singularity Proving Ground – SimAI is applied to any system/mission parameters
- Universe Machine
- Evolution Machine
- Interstellar Technologies
- Gaia Machine
- Brute Force “conscious” entity for Ship AI
- **Save the World**

Project Off Planet Robotics

3 operational areas of development
- Materials mining and processing
- Sheltering and bio-generator construction
- Maintenance and crop raising in within the bio-generators

Build up of the initial operations
- Phase 1 - Minimum seed units and seed size to get materials to micro smelter level
    - Different materials than later versions for full scale operations
    - Keeps sending seed units to new sites.
- Phase 2 - Operations just to keep building printers and mining/processor units and full scale smelting
- Phase 3 - Sheltering and bio-generator construction and maintenance drones - these printers and constructors might even be a roving factory moving between sites using materials processed by the stationed operations
- Phase 4 - Shuffling of units to new sites for new resource locations
- Phase 5 - Far ranging resource grabbing - specialized operations become the source of particular materials.
- Seeding units (printers, excavators, processors). Initial start up entities. Only about proliferating robotic units until enough to start making bio-generator and sheltering construction robotics and maintenance drones.
    - Size of units. Bee swarm versus mega machines versus a balance of all sizes. Efficiency to be found in simulations and real world testing.
    - Bee swarm is incremental gathering and processing that might be useful for building up from small scale to efficient scale - small seeding scenario to get large production quickly but starting from smallest possible seed strategy. A faster way to proliferate into the solar system. Far Out There Note - seeding units to get sheltering and bio-generators going at new solar systems by sending small units ahead with little energy, faster.
    - Small and tiny scale real world tests on Earth will have application if these scenarios prove out in the solar system expansion simulations.
- Self-evolving robotics . Along with constant trait drifting -- Cross breeding. Mixing of components, cross breeding between AI operation cells
- Universal robotic components
    - Arms, legs, footpads,wheels, belts, tracks (armored tank style), tracks and rails for movement of debris and resources above and below

    - Bones - components that will have little wear and tear that can be used for centuries as needed in universal drone schematics
- Missions - types of robotic units - Search, map, resource locations, sheltering locations, excavation, mining, processing materials. Construction, bio-generator production and maintenance, placement of printers, human welfare in bio-generator.
- Cost of materials always evaluated against hardiness - some components are bones, some are recyclable - evolution routines on these criteria.
- Material hours to produce a unit
- 3D printed circuits do not have to meet consumer standards - oversized may be better for printed circuits etc in off planet conditions
- Human scale machinery is a paradigm that must be eradicated for design of initial machinery. Small is a useful scale for for development of SimAI and testing
- Dust Control for robotics - rails to extract rubble, small rails that are reusable for bio-generator structure. Rails could be extremely small. Dust control would be another element in the self evolving systems of SimAI.
- 2 dust control strategies. One for machine wear and efficiency and one for human habitation.

Off Planet Printers

- Several specialized to different materials and productions. Some for adding excretives to sheltering walls. Some for fine electronics or oversized components since the size of motherboards etc. may be oversized for durability and miniaturization may not be an issue.
- Printers include separate Builder robotics to construct the machine parts.

Local Shufflers - attain parts and bring them to a builder and bring processed resources from supply to the printers.

Transporters - heavier shufflers to move robotics, move printers, supplies, energy routings

Resource Scouts - propulsion and/or surface versions

Excavators - For mining materials and forming underground sheltering for bio-generators Excavators can link for leverage of more weight base to leverage

Bones - permanent parts that have infinite usage. Armature and smallest components that can be used repeatedly in various combinations and future robot lineage.

First Robotics Actions

- Backyard and college clubs and simulations to work out robotics pacing - miniature level robotics and command software
- SimAI proving grounds and annual contest such as the battlebot societies with continual feedback loops among collegiate levels

To add your input to the first prototype please go to
https://leapspecies.space/brainstorm-simai-projects