1 Million Module March

Scalable locomotion solutions need to be parallel, decentralised and irrespective of the obstacle. A modular robotic abstract model is used to begin work on the problem. The model is as follows:

![Slide and Rotate](image)

**Figure 1**: Basic motions which the abstract model can perform.

This is the basic building block for a modular robotics solution. There are now several important questions: How to allow movement at the same time? What is a good final goal shape? How to solve the path planning problem on a module scale? How to guarantee that the robot will remain completely connected during operation? How to do all this while allowing each robot to make an independent (decentralised planning) decision?

1.1 How to maintain connectivity

Model the system as a graphical model. Now, find the maximal set of modules which can be moved without disconnecting the graph. Rob has found that the connecting cycles (i.e. the loops which must not be broken to maintain connectivity) are usually small. Using Dynamic Programming (DP) the problem can be solved in a decentralized and scalable way. When the connecting cycles are identified, the modules are locked together to ensure connectivity.

1.2 Simulation

The simulation contains a million modules. Each side of a cube which is unoccupied by another cube is modelled as a state and DP is used to assign that state an L1 distance metric to the goal. Each module, if it is not locked, will then attempt to move the square closest to its goal.

1.3 Discussion

- In this sim, some part of the goal must always be touching the edge of the robot, otherwise the L1 norm cannot be computed. To move the robot the goal position must continuously move
- By modelling the valid states as those empty squares adjacent to the block, the robot is able to traverse over complicated objects.
- To prevent deadlock, two robots are not allowed to lock each other.
- Using a heuristic to speed up the DP algorithm, i.e. Begin from goal and work backwards
- System uses no common memory, only a lot of local communication.
2 Self-reconfiguration Planning in Modular Robotics

Presented by Rowan McCalistair

Using a simulation of the dynamics of a real world modular robot, how do you change shape with the minimum number of steps? There are a number of solutions, but finding the shortest solution is hard, as the state space is extremely large. A common approach is to use a good heuristic, unfortunately it is difficult to envisage a good heuristic for non isotropic 3D models.

A good simplification is to allow only one module to move at a time to reach the desired goal, reduces state space for each step to only 12.

It is however possible for robot to get into a "stuck state", which is where the module cannot move further without assistance from another module. Thus each module needs to be able to ask another module for help, this increases the state space to 17 and adds the requirement that each module must be able to lift itself and one other.

2.1 Module State

The module state consists of:

- Current parts position
- Current direction
- No of d.o.f. for each part (note that some states are symmetric and can often be ignored)

2.2 Discussion

Future work may be to use DDF to track the centre of gravity of the robots convex hull and its centre of gravity. Thus allowing the robot to "balance" itself.