Stair Climbing Vacuum

[Background](http://docs.google.com/index.html) [Use Cases](http://docs.google.com/use_cases.html) [Requirements](http://docs.google.com/requirements.html) [High-Level Design](http://docs.google.com/design.html) [Modules](http://docs.google.com/modules.html) [Design Rationale](http://docs.google.com/rationale.html) [Conclusion](http://docs.google.com/conclusion.html)

High-Level Design

The idea of a robot that climbs stairs is something that is incredibly complex. Trying to replicate the movement of human legs is a daunting task that even our most ingenious engineers are working on. Robots that climb stairs will be a huge step in making robots that can walk around, that can look like humans, and then will be able to do human tasks. If you search for autonomous vacuums that can climb stairs you find a plethora of ingenious labs that are currently working on the problem. They know that, according to realtor magazine 58% of homes are multi story homes versus 42% that are single story. There are plenty of labs like Playupus that are working on different compact stair climbing mechanisms. Their prototypes focus on a spring forward design (See Figure 4).

 Figure 4. Autonomous stair climbing model from design from Playupus

Instead of looking at the lift like a human limb, we instead wanted to focus on a design that would work like a bug with multiple legs. We broke down the main focus of this exploration into getting up stairs and then getting down the stairs. We realized that we needed to build a system that reacts and has a memory. We settled on a table driven interpreters.

To tackle the first problem is important to figure out how to allow the autonomous vacuum to go up the stairs. While the device could have an input of the floor plan of the house, there needs to be a place to input the height of the steps in the house. The plan for this problem is to get the autonomous vacuum to the steps to lift, the height of the steps and inch forward individually lifting one of its legs at a time to get on the steps. This system will be repeatable until the robot hits the top of the steps, which it will know because there is no longer anything in front of it.

Now to tackle the second part of the problem is to include in the robot an infrared detector all around. Allowing the robot to detect when it has hit a ledge, once the robot hits the cliff, the infrared detector will be able to tell the autonomous vacuum how far the cliff is, acting the same way as any infrared motion detector on cameras. As long as the ledge is the top of the stairs that has been previously programmed into the robot will move one leg at a time as it descends until it hits the bottom of the steps and continues to clean the house.

Since we realized that this design is going to require four components that come with the interpreters, we settled on an Abstract Data Type (ADT) solution:

* The first module is going to be the engine that does the work or in our case the autonomous vacuum.
* The second module is going to be the memory that has a pseudocode to be interpreted, in our case the programing of the layout of the house floor plan and the place that it’s cleaning.

The representation of the control state of the interpretation engine or what the system needs to be doing when climbing the stairs. Finally a representation of the current state of the program being simulated, the actual climbing of the stairs as programmed. We need the architecture of the design to be an implicit invocation design. This is required since the complexity of the situation we needed to be sure that not each part of the robot was reacting to just the main input individually. We found that the best situation was a High-Level Design in which the programing communicated in a specific way. (See Figure 5)

 Figure 5. Interaction of the autonomous vacuum climbing and descending the stairs

For a better understanding on how the autonomous vacuum is going to climb or descend the stairs. The following section is going to show a breakdown of each step the autonomous vacuum makes. The flow of making sure that each roller is on the steps it’s working to climb is the key to successfully make sure that the step stays balanced. For a best organized system flow for the implicit invocation system, we build two Unified Modeling Language (UML) charts.

Going up the Stairs

The following breakdown is all the decision making and workflow to allow the autonomous vacuum go up the stairs:

1. When first purchased, user or technician inputs floor plan and height of steps.
2. System is activated remotely or manually by the user.
3. System detects the battery level:
   1. If the battery level is 10% or lower notifies the user on docking station and mobile app.
      1. System does not activate, or returns to the docking station.
   2. If the battery is 30%-11% notifies the user on the mobile app and on docking station.
      1. Continues normal floor cleaning functions until battery hits 10% then activates steps 3.a
4. System detects vacuum bag level:
   1. If the bag level is 85% full or higher notifies user on docking station and mobile app.
      1. System does not activate, or returns to the docking station.
   2. If the bag level is 70%-84% full notifies user on the mobile app and on docking station.
      1. Continues normal floor cleaning functions until the bag level hits 85% then activates steps 4.a
5. The system cleaner comes to where it believes the bottom of the stairs are.
6. The system raises up on the six wheels to the height pre-programed.
7. Detects if it is able to move forward:
   1. Detects if steps are not detected vacuum returns to cleaning.
8. Lifts and folds first wheel in the sequence of 6.
9. Moves forward until wheel is on step.
10. Lifts next wheel in sequence.
11. Repeat steps 9 and 10 until all six wheels are on the step.
12. Detects if there is a next step:
    1. If next stairs system repeats steps 7 - 11.
13. When no more steps the system resumes all regular autonomous vacuum functions and continues.

For a better visualization, we can follow the decision making process with the following UML:

 Figure 6. High Level Functional Overview

Going down the stairs

The following breakdown is all the decision making and workflow to allow the autonomous vacuum go down the stairs:

1. When first purchased, user or technician inputs floor plan and height of steps.
2. System is activated remotely or manually by the user.
3. System detects the battery level:
   1. If the battery level is 10% or lower notifies the user on docking station and mobile app.
      1. System does not activate, or returns to the docking station.
   2. If the battery is 30%-11% notifies the user on the mobile app and on docking station.
      1. Continues normal floor cleaning functions until battery hits 10% then activates steps 3.a
4. System detects vacuum bag level:
   1. If the bag level is 85% full or higher notifies user on docking station and mobile app.
      1. System does not activate, or returns to the docking station.
   2. If the bag level is 70%-84% full notifies user on the mobile app and on docking station.
      1. Continues normal floor cleaning functions until the bag level hits 85% then activates steps 4.a
5. The system cleaner comes to where it believes the top of the stairs are.
6. The system uses infrared systems to detect if ledge is top of steps.
7. Detects if it is able to move forward:
   1. If steps are not detected vacuum returns to cleaning.
8. The system moved forward until first leg is over the ledge.
9. System lowers and unfolds first wheel in the sequence of 6.
10. System lower wheel until it detects that the pressure from the wheel hitting the step.
11. System moves forward until next wheel has nothing underneath.
12. Unfolds next wheel in sequence until it hits the next step.
13. Repeat steps 11 and 12 until all six wheels are on the step.
14. System lowers completely.
15. Detects if there is a next step:
    1. If next stairs system repeats steps 8 - 13.
16. When no more steps the system resumes all regular autonomous vacuum functions and continues.

For a better visualization, we can follow the decision making process with the following UML:

 Figure 7. High Level Functional Overview