TubeBot: a Novel Robot for Scaling Buildings and Providing Floodlight

https://github.com/ElliotWeiner/TubeBot

Elliot Weiner
dept. Mechanical Engineering
Boston University
Boston, MA

Anthony Southmayd
dept. Mechanical Engineering
Boston University
Boston, MA

Parker Boss
dept. Mechanical Engineering
Boston University
Boston, MA

Christina Cavalluzzo
dept. Mechanical Engineering
Boston University
Boston, MA

***Abstract*—Urban environments face challenges in providing lighting in hard-to-reach or temporary setups. This paper introduces a novel solution: a mobile, autonomous floodlight robot designed to scale buildings and provide adaptive lighting. This system integrates reinforcement learning and a robust mechanical design featuring suction cups and stepper motors to navigate and adhere to vertical surfaces. The MonsterGlow robot addresses critical gaps in emergency lighting, construction zones, outdoor events, and security applications by offering dynamic and rapid deployment capabilities. Unlike traditional floodlight solutions and existing climbing robotics, our approach uniquely combines autonomy, adaptability, and sustainable lighting solutions. A simulation environment was developed using CoppeliaSim and ROS to validate the robot's climbing mechanics, path planning algorithms, and lighting performance. Initial results demonstrate the system's ability to learn with a custom learning algorithm and potential for future effectiveness in providing urban lighting through robotics.**

I. Introduction

Urban environments face many challenges with efficient lighting and emergency scenarios. Many hard-to-reach areas are underlit, making traditional infrastructure useless. Temporary lighting is preferred in many situations, such as outdoor events or emergency scenarios. MonsterGlow Robotics is a startup that designs a mobile floodlight robot, revolutionizing urban lighting through robotics. This report will explore some background of related works, then delve into the hardware and software of the system, and finally show how the robot has performed through simulations.

The goal of this startup is to design a groundbreaking robot capable of scaling the side of a building, navigating complex environments, perching atop buildings, and illuminating the ground below with a floodlight. The robot uses advanced environment navigating algorithms and machine learning to climb buildings. This, was combined with hardware that allows the robot to move linearly and rotationally and adhere to buildings with suction cups. When the robot is perched on the upper side of a building, it is a versatile floodlight, lighting up streets, emergency zones, and event spaces.

The target application of the MonsterGlow robot is to provide lighting in urban environments. This includes lighting in hard to reach areas, emergency response scenarios where more lighting is required, and outdoor events where temporary lighting is preferred. It can also act as portable light towers used in construction zones. The robot can also be used in security applications, providing adaptable and mobile lighting where traditional lighting does not make sense.

The world needs the MonsterGlow robot. This robot reduces the need for large infrastructure, creating a sustainable solution and energy efficient urban lighting. In under lit and hard to access areas, this robot offers easy lighting, improving safety and visibility. Because the robot can be deployed quickly, it is perfect for emergency situations and rescue operations that require more lighting. The MonsterGlow robot combines automation and design making it one the forefront of innovation.

II. Related Works

The only existing solutions for flood lighting rely on traditional infrastructure such as flood lights available from The Home Depot [1] and Amazon [2]. These options allow for permanent flood lights in fixed locations however they require human installation and are not ideal for many situations such as hard to reach urban areas, emergency response zones, and temporary installations. Because these options are permanent, they are unsuitable for dynamic environments or situations where rapid deployment is crucial.

In robotics, there are some technologies that share similar autonomous qualities, but they do not fully address the combination of urban lighting, climbing capabilities, and autonomy. Spot, developed by Boston Dynamics, uses advanced autonomous navigation in complex environments. Spot's versatility is strong in mobile robotics, but it lacks lighting qualities integrated with climbing [3]. Another robot is Rufus, a semi-autonomous roofing robot designed by Renovate Robotics. Although Rufus incorporates innovative navigation and robotics, it relies heavily on human help and lacks the autonomy to be incorporated in urban settings [4].

Researchers at the University of Utah have developed a robot called ROCR, a robotic climbing robot. This robot is designed to ascend vertical surfaces using a swinging momentum, however it is designed for security applications and lacks integration with lighting design [5]. Unlike existing options,

the MonsterGlow robot uniquely fills the gap by combining the capabilities of climbing robots with its unique suction cup design and adaptable lighting solutions. It relies on autonomous operation and requires minimal human interaction.

Our approach is unique and in demand because it can attach to any glass surface of a building and provide additional non-permanent lighting solutions. The design also has a lot of flexibility in its movement, so if the light is not directed to the correct spot, then the robot can self-adjust to improve. This is an ability that traditional spotlights lack.

III. Methods

Mechanical Design

The design of the robot is meant to be as simple as possible while allowing for the most flexibility in the movement of each foot. In order to do this, the robot was designed symmetrically with the same mechanisms on each ankle. The midsection of the robot can flip completely over each foot enabling much more range in movement. In each ankle, two stepper motors were used to drive each of the joints. The four stepper motors were attached to 16 tooth 0.8 module gears to 96 tooth gears at a 1:6 ratio, increasing the torque output. The motor that was selected was a NEMA 23 stepper that could rotate at 400 RPM and had a maximum holding torque of 198.2 in-oz. [6] At a maximum extended length of 24 inches, this stepper motor and gear ratio provide enough power to rotate. One of the two ankle assemblies is shown below in Figure 1.

In addition to the movement of each foot, the body of the whole robot can extend outward to have a larger working area. This is done with the use of a NEMA 14 stepper motor with linear actuator. This stepper motor has a thrust load capacity of 28 lbs, which is more than enough to extend the body. This gives an additional 6 inches of reach. [7] This is shown in Figure 2 in addition to the overall design of the robot. Each structural part of the robot was designed to be 3D printed for the initial prototype. Using 3D printed parts, the robot can be quickly changed and iterated upon, as well as giving flexibility in material and infill to either strengthen or lighten each part as needed.

The MonsterGlow robot is able to grip onto flat surfaces using suction cups powered by compact linear solenoids as shown in Figure 3. The solenoid is designed for continuous use with a power rating of 20 W and operates with a push/pull configuration [8]. Each leg has one solenoid and one suction cup, connected by a custom 3D printed connector piece allowing the solenoid and suction cups to be moved simultaneously. The suction cup is a push to grip lifter made of rubber [9], and when the solenoid is at its push configuration, it activates the push on the suction cup. This provides a strong seal between the robot and the surfaces it interacts with.

System Assumptions

Due to the function of this robot being in an outdoor environment potentially high up a building attached to the glass, it has limited deployment scenarios. It is assumed that the robot would not be used in adverse weather conditions.

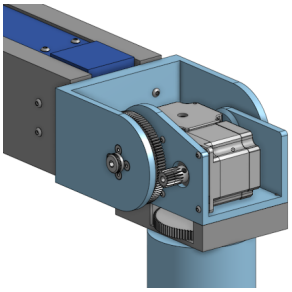

Fig. 1. Angle Design with Two Steppers

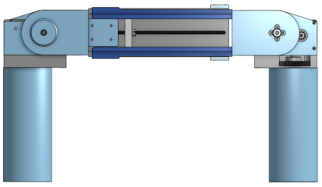

Fig. 2. Overall Design with Linear Actuator Visible

These conditions would include high wind scenarios, rainy or snowy weather. This limits the overall function of this device, but ensures the public's safety by removing the robot from potentially creating a dangerous situation. In addition to the assumption of good conditions, the robot needs a relatively clean surface to walk on. The primary target surface that it would walk on is glass, but that glass needs to be relatively clean so that the robot can find an optimal path to move on and attach to.

Autonomy Stack

The software infrastructure was split into 3 key parts: simulation, affordance, and reinforcement learning. Simulation is described in depth below and represents the connection between our robot's 'brain' and 'body. Affordance and learning make up the intelligent part of the system, which incorporates both model-based and learning-based methods. This section will explore the use of affordance and learning as elements of our architecture.

After examination of the problem itself, it becomes immediately clear that simple point-to-point navigation is not sufficient to traverse two-dimensional environments. Suction hazards such as liquid, dust, and debris could cause catastrophic failure to either gripping ability or the robot as a whole. To avoid this, an affordance algorithm was implemented to identify 'affordable' regions. By passing this information along with start and end point data as network inputs, a policy was learned to identify optimal control values (theta-angle of foot and length-extended size).

Affordance, being a measurement of how 'interactable' an environment is, was calculated on a pixel-by-pixel basis from an image taken from the base of the robot's foot. As seen in Figure 5, the image was processed through a five part pipeline: grayscale, edge detection, variable-radius regional thresholding (VRRT), identity scaling, and reach gating. Grayscale,

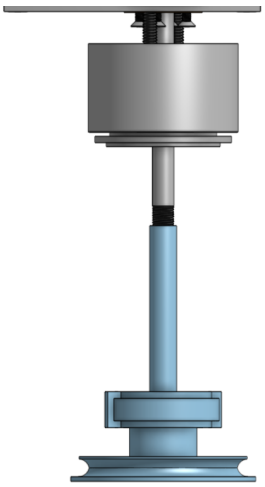

Fig. 3. Linear Solenoid with Suction Cup

used as an intermediate step, allowed for a custom edge detection algorithm to be implemented. This edge method summed the absolute value of pixel intensity differences in all four primary directions (up, down, left, and right). VRRT saw these edges summed over various areas. Larger areas correspond to higher values for a center pixel. Identity scaling sees this map transformed to reduced intensity values toward the left, right, and bottom edges. Reach gating adds a final layer to this, reducing any unreachable location's intensity to zero. To accomplish this, it uses a precalculated projection of camera points onto the two-dimensional environment, as shown in Figure 6, which is then transformed to match the robot's current pose.

Initialize environment
Initialize model
Initialize losses as an empty list
while $N \leq$ iterations **do**
  for $i \leq$ steps **do**
    $robot\_state \leftarrow$ environment
    $affordance \leftarrow$ affordance($robot\_state$.image)
    $actions \leftarrow$ model($affordance$)
    calculate reward
    calculate loss
    append *loss* to *losses*
    $i \leftarrow i + 1$
  end for
  $mean \leftarrow \frac{1}{|\text{losses}|} \sum \text{losses}$
  backpropagate
  $N \leftarrow N + 1$
end while

Fig. 4. Training Loop

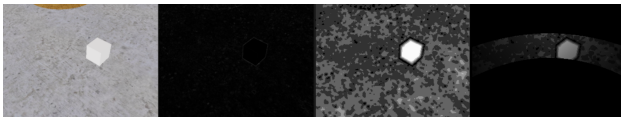

Fig. 5. View from Simulation

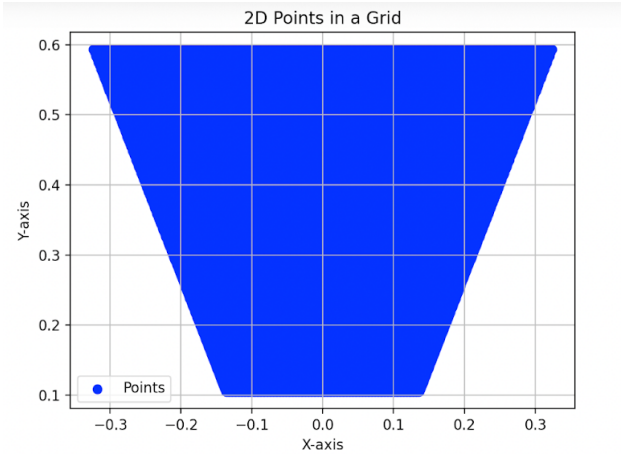

Fig. 6. Pre-calculated Mapping from Pixel Space to State Space

Simulation

As an alternative to implementing the MonsterGlow robot with hardware, a proof of concept simulation was created instead. The software that was chosen is CoppeliaSim, formerly known as VREP. CoppeliaSim features built-in dynamics simulation, the ability to import and assembly custom models, and built-in Robot Operating System (ROS) integration. The dynamic portion of the simulation model, as seen in Figure 5, defines the surfaces that will have collisions with the environment. There is an additional visual model, seen in Figure 6, that doesn't contribute anything to the simulation besides a "pretty" factor. It should be noted that, while the model shown in Figure 6 is an early prototype, the dynamics of the robot are accurate to the final design and the decision was made to not spend time updating the "pretty" model. The environment, and contained obstacles, can be seen in Figure 7. The floor of the environment features four types of flooring to simulate different surfaces and prevent the machine learning model from learning only a specific surface. The obstacles are entirely 2D and they are randomly oriented and colored to simulate a variety of "dirty" spots. The simulation was run in the horizontal orientation, as opposed to vertical like a wall, due to convenience for human interpretation.

Input from the machine learning portion of the code is sent to a python object that acts as a virtual twin of the robot. This twin then publishes joint values and suction cup flags through ROS into the Lua backend of the simulator. Each joint in the simulator will receive the target position and uses a built in PID controller to move to the position. Each PID controller is roughly tuned so that the robot moves quickly to the target configuration with minimal oscillation. However, the

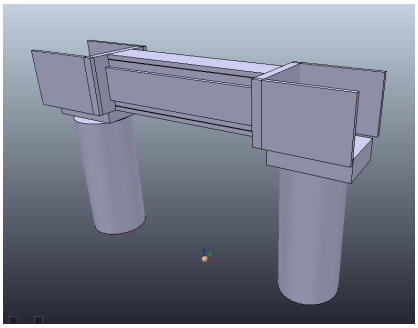

Fig. 7. Dynamic Model

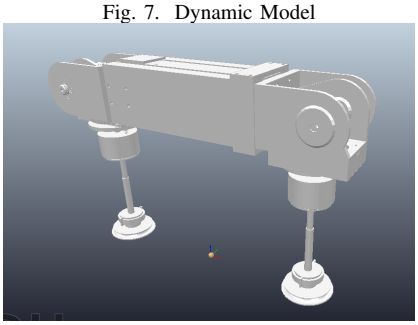

Fig. 8. Visual Model

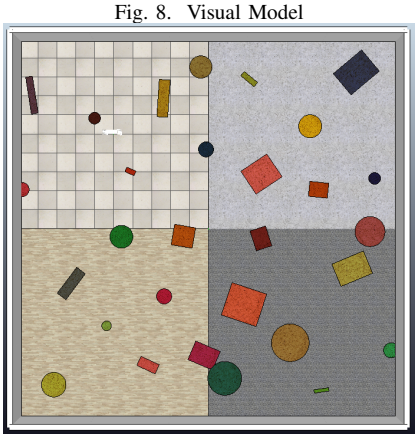

Fig. 9. Environment

PID controllers were not the main focus of the project and thus they should not be considered fully tuned. The base of each leg has an RGB and depth camera embedded in the center. These cameras send their images out to the machine learning model through a ROS topic. The depth cameras were unable to be configured to send raw distances and instead send scaled relative distances. For this reason, the obstacles were limited to 2D because an accurate method of measuring the environment geometry was unavailable. As a final note, due to the ROS integration, the simulated robot can be quickly swapped out for a hardware implementation in future iterations.

Reinforcement Learning

Reinforcement learning was an obvious choice to balance advancement towards a goal and regional avoidance. We decided on a model that would predict length (extension) and theta (turn) values for a robot, given that it was already pointing at a goal. The simulation capacity of this project allowed for repeated testing and therefore self-supervision. After setting the groundwork for learning, model structure and reward become the two most important elements of finding an optimal policy. Because theta and length are generally continuous, an Actor-Critic method seemed to be the best choice. In a twist though, we realized that by back-calculating affordance from our precalculated map we could definitively know the reward for a given state. By choosing to optimize only for single steps, the critic was replaced with a ground-truth reward. The model itself is constructed from a data encoder-decoder architecture. In our experiments, this allowed for simplified swapping of components. We also used epsilon-greedy methods to explore the image-space (a subset of the state space) and tapered it off over time to focus on exploitative actions.

Reward, being a key part of unsupervised learning, was one of the most important defined parameters in the network. By calculating it in terms of advancement towards a goal and affordance, we provided a way to guide the network to optimal outputs. Equation 1 outlines a loss function based on this reward. Advancement was equated to the previous distance to goal minus the current distance to goal, where the advancementmax was the maximum length of the robot (or the distance to goal if it was smaller). Stridemax was the maximum length of the robot, to account for steps away from the goal. Affmax was the maximum affordance in the map.

$$\frac{advancement_{max} - advancement_{actual}}{advancement_{max} + stride_{max} + e} + \frac{aff_{max} - aff_{max}}{aff_{max} + e} \tag{1}$$

IV. Results

We measured our robot's success based on its overall training loss over time. As this paper is in many ways an exploration of viability for reinforcement learning and calculated affordance, charts displaying the change can provide valuable insight into the effectiveness of our solution. We conducted two experiments with different solution scales to prove the training method and see the immediate success of our design.

Our first experiment consisted of a scaled-down version of our model, specifically to test our method of learning. To do this, we removed the affordance entirely from the state space. We also set length to be a static value, which limited our action space to only output a theta value ranging from 0 to 2π. Our reward function was similarly constrained to only account for advancement. We used 3 fully connected layers and no data encoder (as there was no image) for our model. This configuration, in theory, should have led to a model which learned theta values which guided it directly towards the goal. Figure 10 displays a clear convergence towards 0.18, which represents the best furthest possible advancement with a constrained length. The oscillatory behavior apparent in the graph makes sense when analyzed in terms of the loss function, as it never quite reaches zero without length

included. From this, we gathered that our method was viable and proceeded to explore a more complex state / action space.

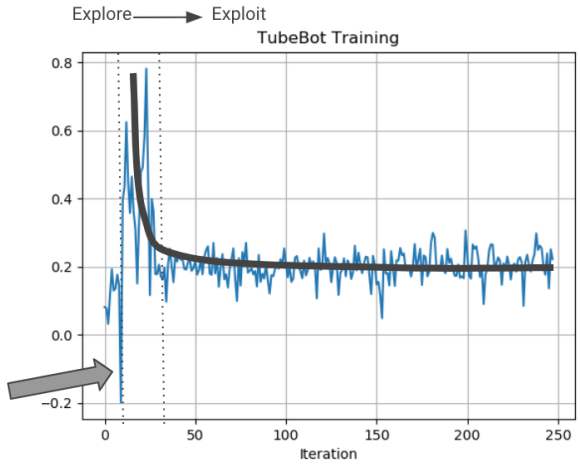

Fig. 10. Experiment 1 - Scaled Down Modeling with No Affordance or Length

The second experiment saw the return of affordance and length to our model. With the added complexity, a convolutional encoder was added in combination with the same decoder layers used in Experiment 1. As seen in Figure 11, it is immediately apparent that the training was relatively unsuccessful despite the increasing the duration by almost five times. Based on architectural and complexity differences, our team inferred this was likely from the added 'noise' of a high-dimensional affordance map. Error was also attributed to errors such as simulation failure, encoder simplicity, decoder depth, and training time.

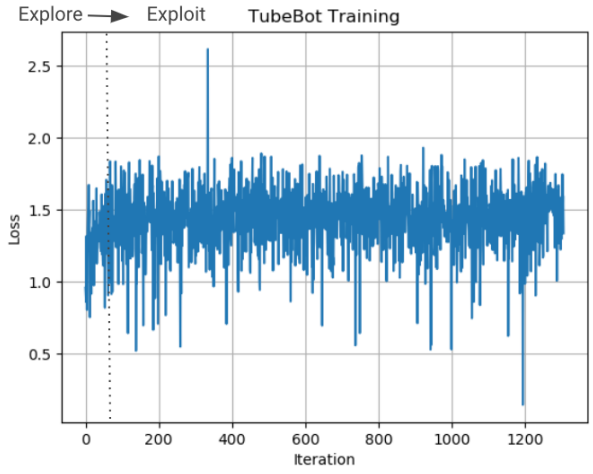

Fig. 11. Experiment 2 - Full Model Training

V. Conclusion

The hardware for the design of the robot is relatively novel and would need to be built in real life to validate its functionality. In the future if the robot was built for production, the main structural parts would have to be adjusted for different, more durable manufacturing methods. This would likely involve injection molding or machined parts. The design could also be further improved with the design of the electrical system being integrated into the structural components.

The software experiments provided a base for future research into robotic step selection. While our method is proven to work in learning basic parameters, it seems to have trouble with more complex problems. State space complexity can cause especially difficult-to-learn policies, which prompts researchers to implement more complex models and longer training times.

Future work for this project is already underway. Many of the issues discovered in Experiment 2 are being addressed to allow for smoother training. To provide better generalization of abstract global shapes, as would be expected in our affordance map, we are working to implement a state-of-the-art Visual Transformer (ViT) in the place of our convolutional layers. Hiera-B has been our choice and thus far has proved to be a viable replacement [10]. Additional fully connected layers have been added to the decoder and simulator issues have been investigated as well. While this model will have a much higher training time and therefore withhold results for longer, inspection of our training appears to show improvements in learning.

References

[1] Home Depot, "Par 180 degree white motion sensor wired outdoor 2-head dusk-to-dawn security flood light." https://www.homedepot.com/p/Defiant-PAR-180-Degree-White-Motion-Sensor-Wired-Outdoor-2-Head-Dusk-to-Dawn-Security -Flood-Light-17000163/324059075, 2024.

[2] Amazon, "Melpo led flood light outdoor, 500w equivalent 5000lm smart rgb landscape lighting with app control, diy scenes - timing - warm white 2700k - color changing uplight, ip66 waterproof us plug (2 pack)." https://www.amazon.com/MELPO-Outdoor-Equivalent-Landscape-Lighting/dp/B0BW8VFN32, 2024.

[3] Boston Dynamics, "Spot product page." https://bostondynamics.com/products/spot/, 2024.

[4] Renovate Robotics, "Renovate robotics homepage." https://www.renovaterobotics.com/, 2024.

[5] W. R. Provancher, S. I. Jensen-Segal, and M. A. Fehlberg, "Rocr: An energy-efficient dynamic wall-climbing robot," *IEEE/ASME Transactions on Mechatronics*, vol. 16, no. 5, pp. 897–906, 2011.

[6] McMaster-Carr, "Stepper motor." https://www.mcmaster.com/6627T242/, 2024.

[7] McMaster-Carr, "Linear actuator." https://www.mcmaster.com/8677N51/, 2024.

[8] McMaster-Carr, "Sealed linear solenoids." https://www.mcmaster.com/products/solenoids/direction-of-operation linear/.

[9] McMaster-Carr, "Suction cup." https://www.mcmaster.com/products/suction-cups/gripping-method suction-cup/, 2024.

[10] D. Bolya, "Hiera repository." https://github.com/facebookresearch/hiera, 2024.