Water Balloon Modification Design

Rebanto Nath & Ian Russell

Launcher Aspect to be Improved: Aiming and using the launcher (human error aspect), while considering external factors such as wind, air resistance, humidity, temperature, etc.

Research:

There have been several projects and methods proposed that are similar to our concept of making an autonomous water balloon launcher. Students at Washington University have created an Automatic Projectile Launcher for a flywheel launcher. The launcher is made up of a Raspberry Pi 3b+, to capture the target using the camera; the flywheel launcher; and a pitch-yaw axis robot with Step and Direction Motors to control the launcher. Specifically, the aspect that appeals to our modification is the image processing done in the Raspberry Pi. In the Automatic Projectile Launcher, the targets are marked on a wall with blue tape. Using OpenCV to identify those points, the launcher accordingly shifts its motors to point in that direction using defined pitch and yaw angle functions. We intend on utilizing such image processing in our modification to identify the target location and calculate distance.

Another similar concept, primarily seen in online video games (First Person Shooters), is aim assist. Aim assist, primarily implemented for players on game consoles playing against PC players, automatically guides crosshairs (to a certain extent) towards an enemy. This is mirroring a vital concept in our modification plan - automatically aiming the launcher towards the target. Traditionally, in online games, the aim assist software has access to the enemy's locations and automatically shifts the crosshair of the player. However, in the context of the autonomous water

balloon launcher, there are many more factors at play. While we know the distance and angle of reference of the target from the camera and user input, variables such as wind speed/direction, humidity, air resistance, temperature also affect the landing spot of the water balloon. Therefore, a separate algorithm or model is required to regress to these non-linear patterns, such as a Neural ODE (Ordinary Differential Equation) or PINN (Physics-Informed Neural Networks). This has been done previously many times, and is a major method for physics simulations. An example of a project is found in this GitHub repository: https://github.com/tbensky/PiNN_Projectile. We plan on utilizing PINNs or some other model to factor in the external variables when calculating the launch angle and pullback for our autonomous launcher.

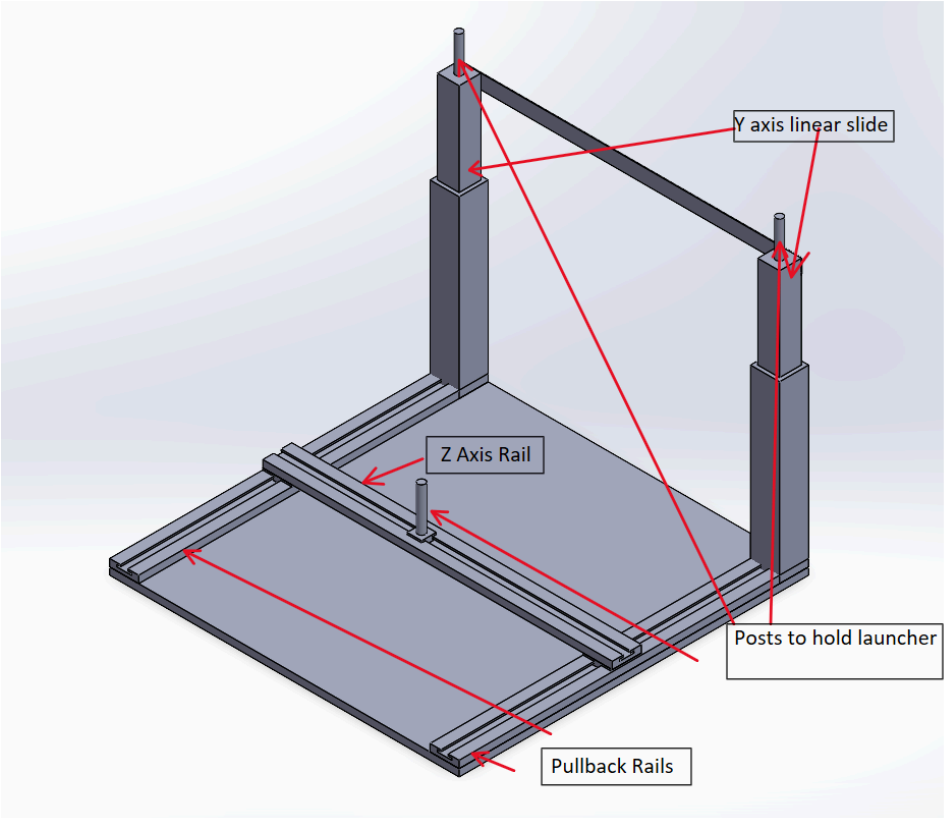
Citations:

https://github.com/tbensky/PiNN_Projectile

https://sites.wustl.edu/automatedprojectilelauncher/hardware-walkthrough-demo/

https://equis.cs.queensu.ca/~equis/pubs/2023/schneider-chi-2023.pdf

Blueprint:



The problem this blueprint addresses is the human error and the accuracy when aiming and allowing the launcher to factor in wind and air resistance and adjust the machine accordingly.

It solves it by having 2 pull back rails that pull back the water balloon launcher and has a z-axis rail that allows it to aim left and right. Once the z-axis and pull back is set the linear sliding posts allow it to adjust the maximum height quantity. To be successful the modification would need to be automated and allow the user to choose where they want the balloon to land and it would automatically adjust the parameters based on our knowledge of kinematics, forces, energy, and momentum. On top of functions based on classical physics, the solution would use a supplemental algorithm such as a PINN (Physics-Informed Neural Network) to account for external variables such as wind, humidity, temperature, air resistance, etc. To be considered

successful on a quantitative metric, the modified autonomous water balloon launcher could be tested in an online simulation (ROBLOX), with an earth-like environment and also tested in the real world (using manual measurements). If the landing spots of these two tests are within an error range of each other over multiple trials, the modification could be deemed successful.

Role of Physics (Kinematics, Forces, Energy, and Momentum):

Many physics concepts are used throughout this concept of an autonomous water balloon launcher. The user initially inputs the target location, so using the camera, the launch can acquire the relative x and y distances to the target location. Then, using the kinematic equations (shown below), the launcher can calculate the launch angle and initial velocity. While the equations below relate velocity, angle, time, and distance, we are not solving for any specific value of angle or velocity. Instead, we will estimate the launch pullback using different convenient angle presets that the launcher is already wired to.

Horizontal: $x = v * cos(\theta) * t$

Vertical: $y = v * sin(\theta) * t - \frac{1}{2}gt^2$

Once the estimated velocity is determined, the launcher then has to determine how much pullback is required to achieve that velocity. This can be done by utilizing energy concepts, specifically relating the initial elastic energy of the launcher band and the kinetic energy of the water balloon after it launches. The below equation, when simplified, offers a formula for calculating pullback (given the spring constant of the launcher) to achieve a velocity.

$$\frac{1}{2}mv^2 = \frac{1}{2}kx_s^2$$

$$x_s = \sqrt{\frac{mv^2}{k}}$$

As mentioned previously, we intend to add an AI correction layer that accounts for forces such as air resistance and wind. While gravity is already accounted for in the kinematic equations, our additional model (Neural ODE or PINN) could account for other forces such as drag (equation shown below). Beyond the classical physics prediction, the AI correction layer, which is trained on such equations, can shift the angle slightly to achieve maximum accuracy.

$$F_d = \frac{1}{2} C_d \rho A v^2$$

Finally, the AI correction layer can also factor for momentum when shifting angle and velocity for better launch accuracy. This is because a higher momentum causes less deviation of the balloon from external factors such as wind. Therefore, if the calculated momentum is low (equation shown below), the launcher can increase the pullback (to increase velocity) so the balloon has a higher chance of staying on course, while also changing the angle to maintain the target landing spot. Furthermore, the calculated momentum can also serve as a confidence indicator. Pairing the momentum value with data from wind sensors can offer an informative confidence value for the user. For example, if wind speed is high and momentum is low, the confidence value is low.

$$p = mv$$