3D Printed Drivetrain Development for Inspection Robot

Elijah Wilkinson

College of DuPage, University of Illinois at Chicago

Abstract

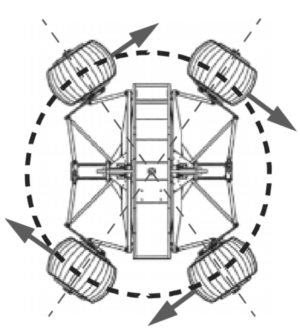
The Remote Viewing Robot (RVR) will reduce radiation exposure to personnel and reduce beam down time by performing tasks in the accelerator tunnels. This paper covers a design update to RVR's drivetrain, swapping out wheels for tank tracks. The tank track links were designed to be 3D printed to allow them to be replaced easily and affordably in the case that they became radioactively contaminated with dust from RVR's operation in the accelerator tunnels. Additionally, software was added to RVR to allow it to be driven remotely over a WiFi network.

Purpose of RVR

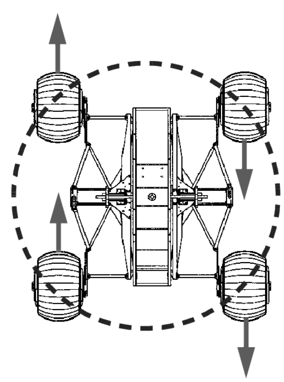
The purpose of the Remote Viewing Robot (RVR) is to investigate Fermilab's accelerator tunnels. It has a modular chassis for mounting different payloads such as inspection cameras, tunnel mapping equipment or radiation measuring devices. The goal is to reduce radiation exposure to personnel where possible and improve beam delivery. One example would be using RVR to locate and determine the cause of a water leak on the accelerator. RVR would be able to enter the tunnel and quickly identify the issue so personnel could swiftly resolve the issue. Without an inspection robot like RVR, personnel would have to wait for radiation levels in the tunnels to subside and create an approved safety plan before entering the tunnel to locate the leak which could take weeks, and in the end, personnel would still be exposed to radiation. Additionally, sending RVR into the tunnel rather than a group of people will free up personnel for more important tasks. The goal of this project is to equip RVR with a drivetrain that will allow it to effectively carry out its tasks. The new drivetrain makes use of inexpensive, rapid-prototyping technologies to reduce maintenance costs. Additional work was done to create and test software to drive RVR.

Past Work on RVRs Drivetrain

RVR is constructed from a welded aluminum frame. It is powered with two motors, one driving each side of the robot, using skid steering. Initially it had a tank tread drivetrain. However, that drivetrain was replaced with four wheels due to concerns with replacement tank track availability and cost. The company that manufactured the tracks no longer existed and even before that, the tracks had been very expensive. This was an issue because the track was likely to collect radioactive dust that could not easily be removed as the belt was a Kevlar fabric based. While the four wheels were much better for radioactive dust contamination, RVR didn't turn well. RVR's chassis is longer than it is wide and uses skid steering, so during turning the wheels had to slide to the side more than they moved forward resulting in poor and jittery steering. While converting to wheels was valuable for learning what works with skid steering, another redesign was necessary to update the drivetrain to steer better, implementing what was learned.



Explicit Point Turn Configuration



Skid Steer Point Turn Configuration

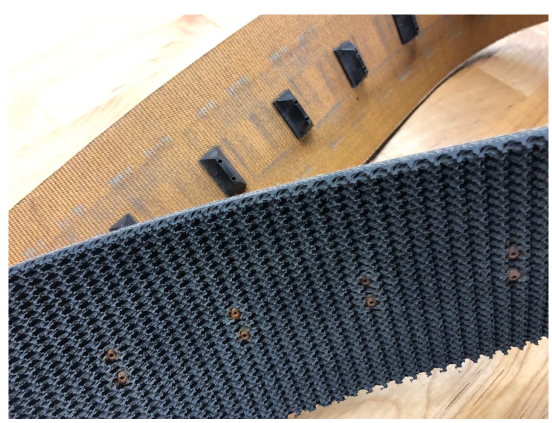


Diagram showing how skid steering with wheels works [1] and photo of original fabric-based tracks.

Design Process

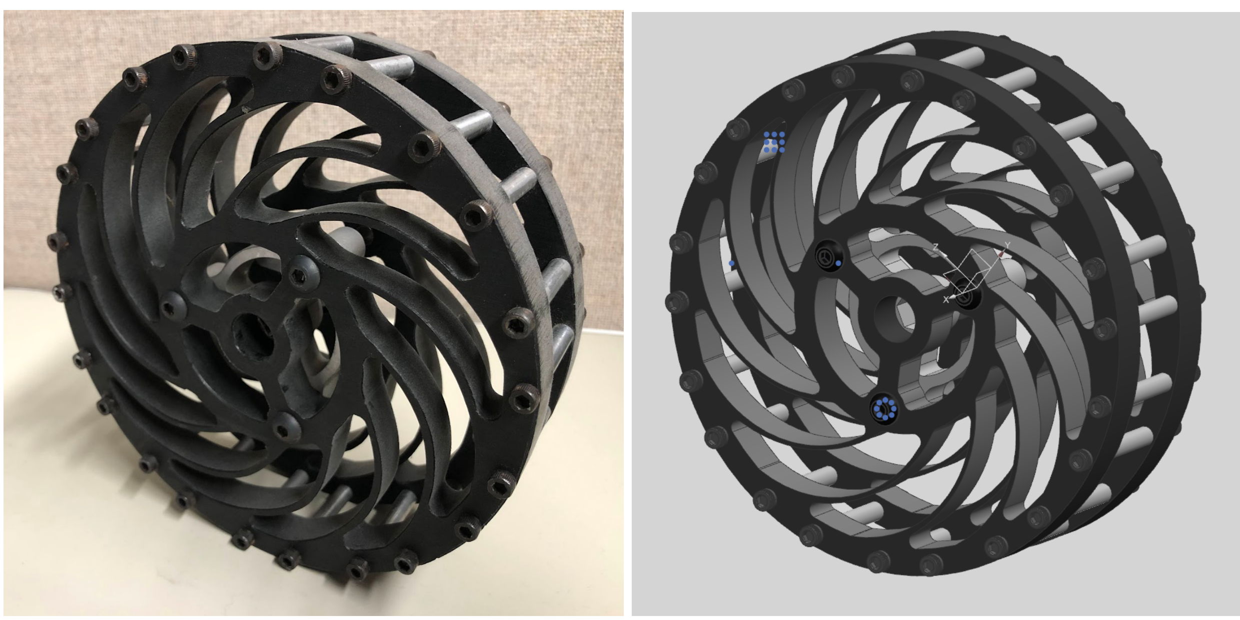
When designing the new drivetrain, moving back to using tank tracks was a preferred solution. Primarily this was because tank tracks were known to work with skid steering and the robot was already set up to use them. Additionally, while adding a different method of steering could make wheels work (such as using mecanum wheels or adding pivots to the wheels for explicit point turning) adding a different method of steering would require major changes to RVR's power electronics including adding more motors and motor controllers. Therefore, a tank tread drivetrain was the ideal solution if one could fit all the design requirements and preferences.

Requirements for drivetrain:

1. The drivetrain should perform well in RVR's operating environment. Good traction on concrete, which may be wet, and precise steering are important.
2. The drivetrain should be easily replaceable for when the drivetrain comes in contact with radioactive dust; wheel treads or tank tracks need to be easy to replace when necessary.
3. The drivetrain should have affordable replacement parts; replacement of contaminated parts cannot be expensive as it will likely be a recurring cost every time the robot is deployed.

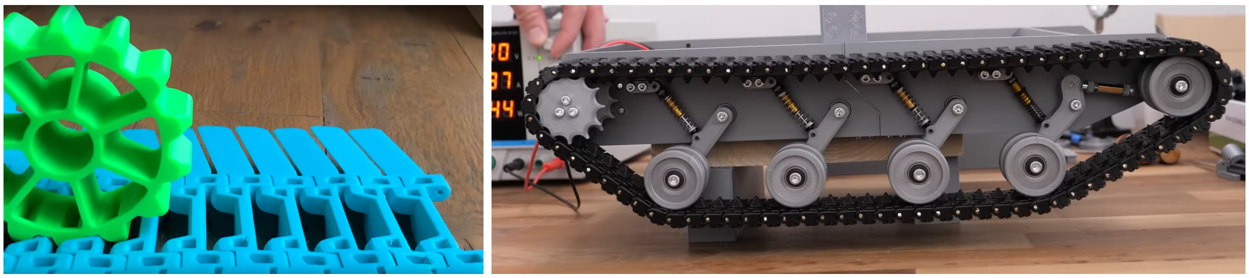
Preferences for drivetrain:

1. It would be preferred if the disposable parts of the drivetrain were 3D-printable. This would allow for rapid prototyping, customizable traction for different environments and most importantly, easy in-house manufacturing of replacement parts.
2. It would be preferred if the drivetrain could utilize existing wheels in the case that tank treads are used. These wheels already work with RVR and would save design time.



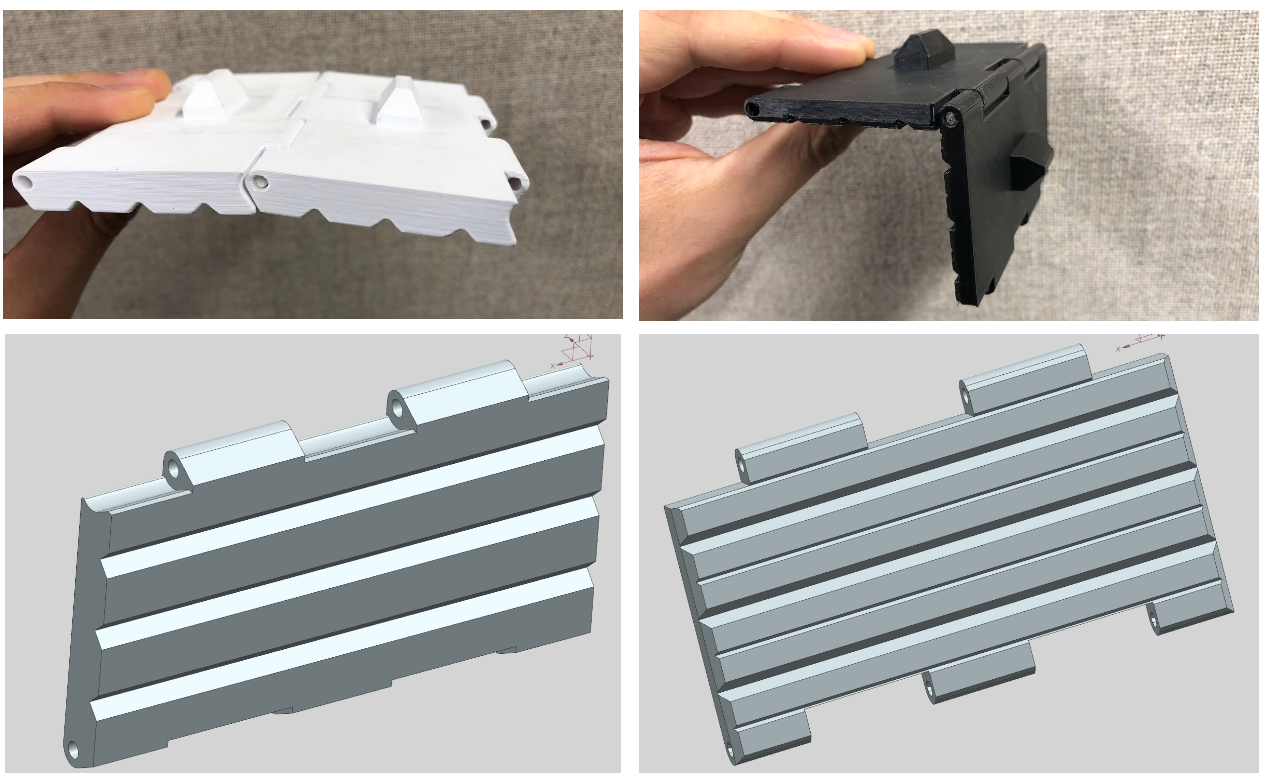
In order to design tracks that fit the editing wheel, a matching CAD model had to be created.

The design process started with researching differing tank track designs created by other people. The most common designs had sections of hard track connected together with a rod at the joint or having the tracks themselves clip together. Initially, the first prototype planned to follow this design using a 1/8in rod as the link between tracks. However, during prototyping this was changed to using 2.85mm 3D printer filament as the joint as it was more than sufficiently strong enough, cheaper and readily available in the robotics lab.



Most other track designs researched had gear-like powered wheels that linked into the track [2], [3].

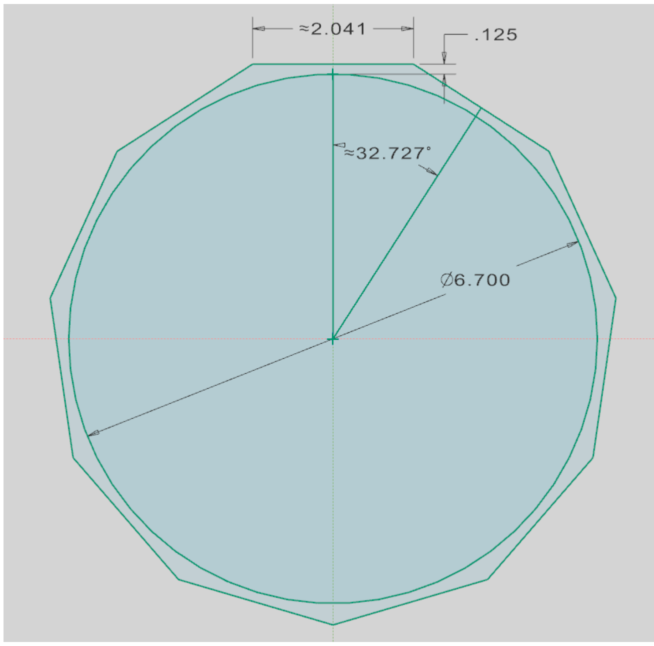
The initial prototypes of the track links were .375in thick, 4in wide and 2.041in from joint to joint and had three small tread grooves per link. The limited number of grooves was so that a maximum amount of surface area was in contact with the ground. Excessive tread was not necessary with RVR's concrete intended driving surface. During the prototyping phase of the design process, the thickness of the track links was reduced to .25in, and the amount that the tracks could rotate was increased to allow a better fit over obstacles.



Several iterations of prototypes (left) lead to the final design (right)

One concern about using the existing track wheels arose from the fact that most of the designs had the track being powered by a gear-like wheel, with the teeth of the wheel hooking into the track links and moving them along. With RVR's existing wheels, the tracks have teeth that hook into the wheel. This caused concern that the wheels wouldn't hold onto the links as well. However, after testing with prototypes, this turned out to not be an issue.

The dimensions for the teeth of the tracks came from the original fabric tracks that the wheels had been designed for and were simply measured with a caliper. The hardest dimension to find was the center-to-center distance for the joints of the tracks. The distance between tracks was estimated by constructing an 11-sided polygon, to match the number of tooth holes in the wheel. This resulted in the joint-to-joint distance of 2.041.



Several materials were tested in the track's prototyping phase, including Polylactic Acid (PLA), Polyethylene Terephthalate Glycol (PETG) and Thermal Polyurethane (TPU). The TPU-printed track links had a much better grip on the concrete floor surfaces but lacked strength. The PLA and PETG links were much more rigid and had less traction on concrete. However, if TPU could be combined with one of the harder filaments it would be possible to achieve a balance of both benefits.

A few experimental prints tested the ability of the single-extruder printers available in the robotic lab to print multiple materials. These met with varying degrees of success. By printing first in TPU and then in a separate print using PLA the two materials could be merged, although the adhesion was not great over small areas. Printing a single part with a filament change part way through was more successful, although it still didn't have great adhesion between TPU and PLA especially if it was on small areas. A possible reason for this could be their different melting temperature, with PLA at 215C and TPU at 240C. When the PLA printed onto the TPU, it didn't heat up the TPU enough to fully bond. However, by using PETG with a printing temperature of 240C instead of PLA, all the adhesion issues were resolved. The PETG and TPU adhered to each other without issues. This is likely due in part to the fact that PETG and TPU both naturally have much better layer adhesion than PLA to begin with.

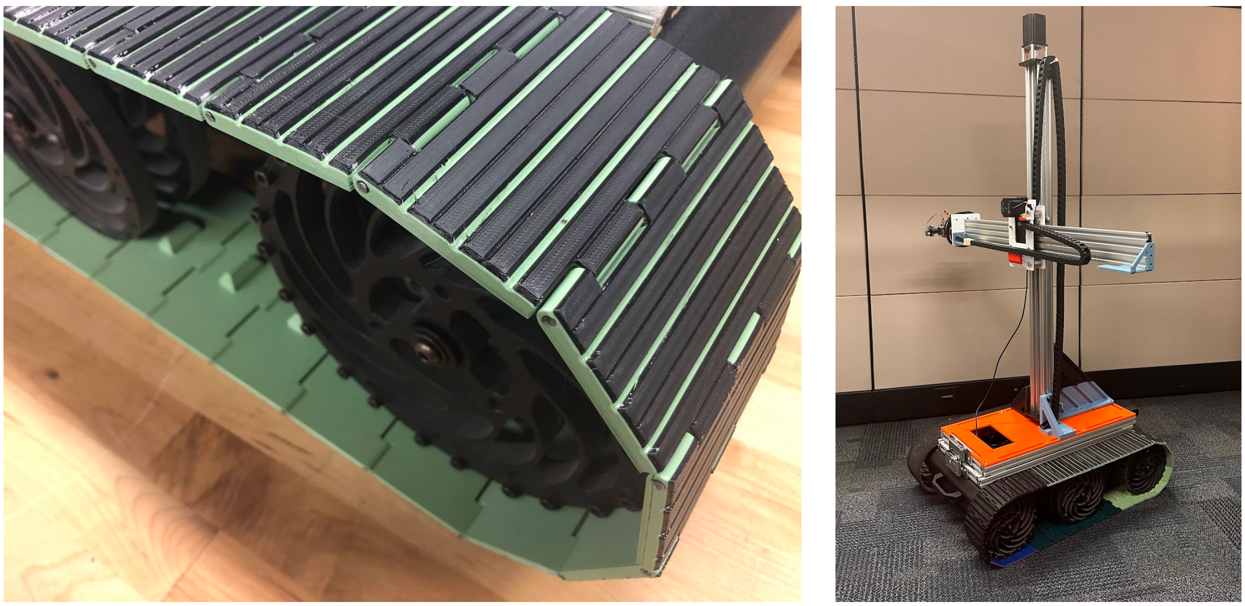
Final Design of Track Links

The final design of the track links used both TPU and PETG combined in the same print using a filament swap. The first 6 layers of the track link's 3D print was done in TPU and the rest in PETG. This allowed for a better grip on the ground while maintaining a strong part. The three tread grooves were slightly reduced in size and miniature grooves were added between them to provide more edge surfaces while still maintaining maximum contact area with the ground.

Each track has 31 links. 6 links can be printed at a time with a print time of 12 hours (approximately 2 hours per link). Using one 3D printer, it takes 3 days to print one track.

Material Use Summary:

- TPU - 6.3g used per link, 195.3g per track, 390.6g per set
- PETG - 21.4g used per link, 662.5 per track, 1324.9g per set



The final tracks are shown mounted on RVR.

Operating RVR

With the new drivetrain complete and installed, RVR's control software required an update before operating. After installing a fresh Raspberry Pi operating system, code was taken from Helpful Robotic MESsenger (HERMES), another robot in the lab. A small modification to the code swapped the network name of the Raspberry Pi to connect to RVR instead of HERMES. After installing the necessary packages to run the robot code (formerly called the server code) and the companion laptop control code (formerly called the client code) RVR was driving.

There were several issues with the program. First, only certain buttons should be able to command the robot, which are:

Motion Commands
w - forward

s - backwards
a - turn left
d - turn right

Other commands:
t - stop
e - exit (end code on Robot side)
l - set motors to a low speed
m - set motors to medium speed
h - set motors to a high speed

Old testing commands (all these now removed)
r - run (seemed intended to make the robot move in whatever direction it had been going previously, but actually only resulted in making the robot crash)
f - forward (slightly different code then w, but with the same effect)
b - backward (slightly different code then w, but with the same effect)

However, when driving RVR, pressing any undefined key would result in a repeat of the previous actual command. This was strange since the code would print out "unknown input" seeming to recognize that it wasn't receiving an actual command.

Additionally, RVR had an issue misinterpreting commands during network lagging. Normally commands came through to the robot as follows:

…
HB *- when not pressing any key a regular Heart Beat (HB) command pluses*
HB
'w' *- when the w key is pressed the 'w' (forward) command comes through*
'w' *- the w common is continually resent to the robot while the key is held down*
'w'
'w'
'w'
't' *- when the w key is released, a 't' (stop) command comes through*
HB *- the heartbeat command continues*
HB
…

However, when the network lagged commands would be received as follows:

…
HB
HB
'w'
'w'
'w''w''w' *- due to network lag, commands are bunched together. The code doesn't interpret*
'w''w' *these commands and just continues the previous command*

‘w’‘w’	
‘w’‘w’‘t’	- *the ‘t’ (stop) command comes through bunched with the previous command and*
HB	*since it is never read, the robot continues driving forward, out of control*
HB	- *heartbeat command continues but the robot will continue driving since it never*
…	*received a stop command*

This was a major issue since when the stop commands were missed due to bunching, RVR would continue driving or turning in whatever direction it had been moving in previously. Since the network lagged out for 5-10 seconds at a time, even pressing the t key to stop was unreliable. Losing control of the RVR for 10 seconds during a mission in the tunnels could be catastrophic so this was a crucial issue.

Fortunately, both the network lag issue and the random key commands were solved by adding code to change the else statement that handled cases when commands were not recognized. By interpreting all unrecognized commands (both from command stacking like ‘w’’w’’t’ and from random keys being pressed like ‘p’) as a stop command this issue was fixed. Now if the network lags out and the commands bunch up, RVR doesn’t move while waiting to receive proper commands.

Remaining Tasks

There are a few tasks remaining to get RVR mission ready. The drive program needs to be set up to start on boot-up of the Raspberry Pi, so that when the robot is turned on the program runs and connects via WiFi to the controlling laptop. Currently the only way to run the program on the Raspberry Pi is using the desktop interface, with a keyboard mouse and monitors. This is fine for testing but impractical for normal operation.

It would be beneficial for operating in tunnels if RVR provided its own ad hoc network that the controlling laptop could connect to. Currently RVR and on the controlling laptop communicate on the Fermilab fgz network which is not present in all of the tunnels and would have to be brought in with an additional router.

RVR’s arm and camera systems, while operational, are not running on the same code as RVR. It is important to get them connected to and operated by the same controller laptop in order to use that payload remotely while also driving the robot.

Finally, RVR should continue to be tested to ensure its long-term reliability. It should also go through training exercises in environments similar to where it might be operating to test the limits of its capabilities and to train the operating team to use it effectively.

References

[1] B. Shamah, Experimental Comparison of Skid Steering Vs. Explicit Steering for a Wheeled Mobile Robot, 1999. (https://www.semanticscholar.org/paper/Experimental-Comparison-of-Skid-Steering-Vs.-for-a-Shamah/bfc41a31333f1bd11759565bd7e63a756d476ae2/figure/18)

[2] R/C Snowcat - New Track Design - RCTESTFLIGHT, 2021. (https://www.youtube.com/watch?v=OXDO4VL4NaA&list=PLXvxJNOIXBsN97d3_b5AXquRSAQOs3tVV)

[3] Fully 3D Printed TANK / Tracked Robot Platform, 2023. (https://www.youtube.com/watch?v=q3XNYwNZ97w)