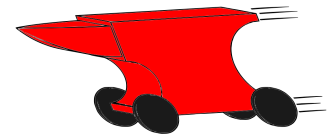
Motion Planning and Control in FTC

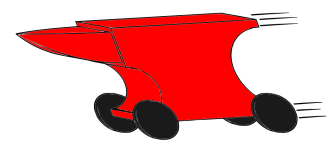
Kelly Muir
Captain, Team 8367 ACME Robotics

FIRST Championship Conference Houston 2019



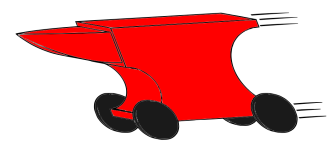
Terminology

- Motion Planning - planning prior to execution of motion



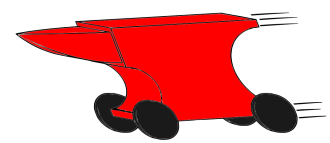
Terminology

- Motion Profile - destribes 1d displacement as a function of time



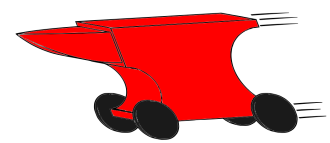
Terminology

- Path - describes a series of positions for a robot as a function of displacement



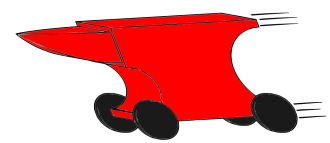
Terminology

- Trajectory - Path + Profile, robot state as a function of time



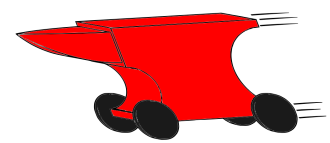
Terminology

- Motion Control - achieving a desired state in a particular system



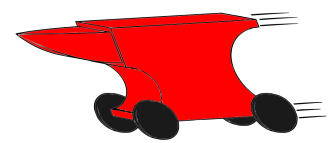
Terminology

- Feed forward control - predictive control based on known system dynamics



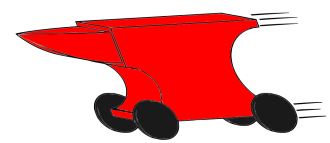
Terminology

- Feed back control - corrective contnrol in reaction to error

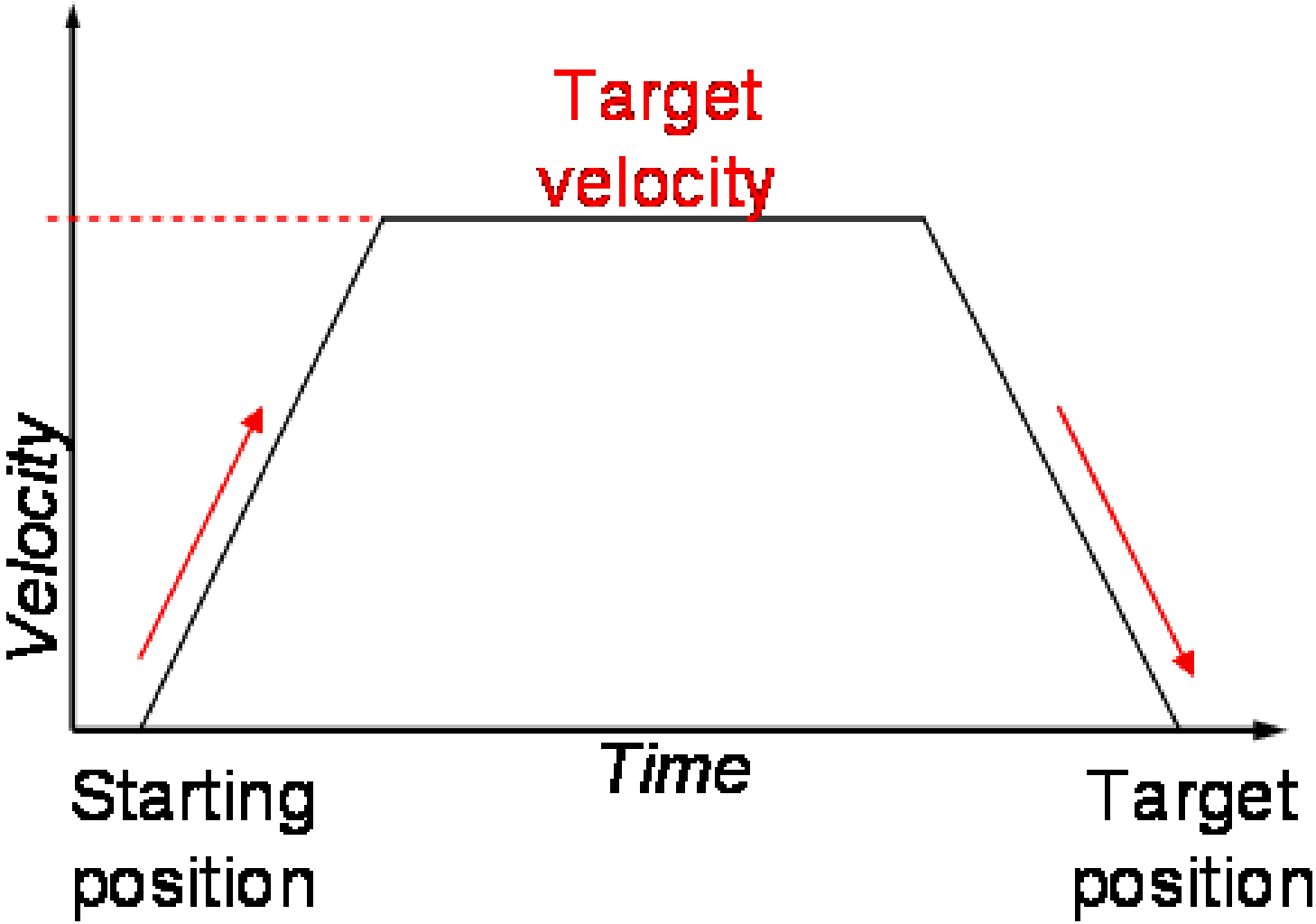


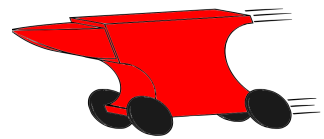
1D Motion Planning

- Current State $\rightarrow$??? $\rightarrow$ Target State
- Constrains:
  - Time optimal
  - Continuity
  - Observe physical limitations

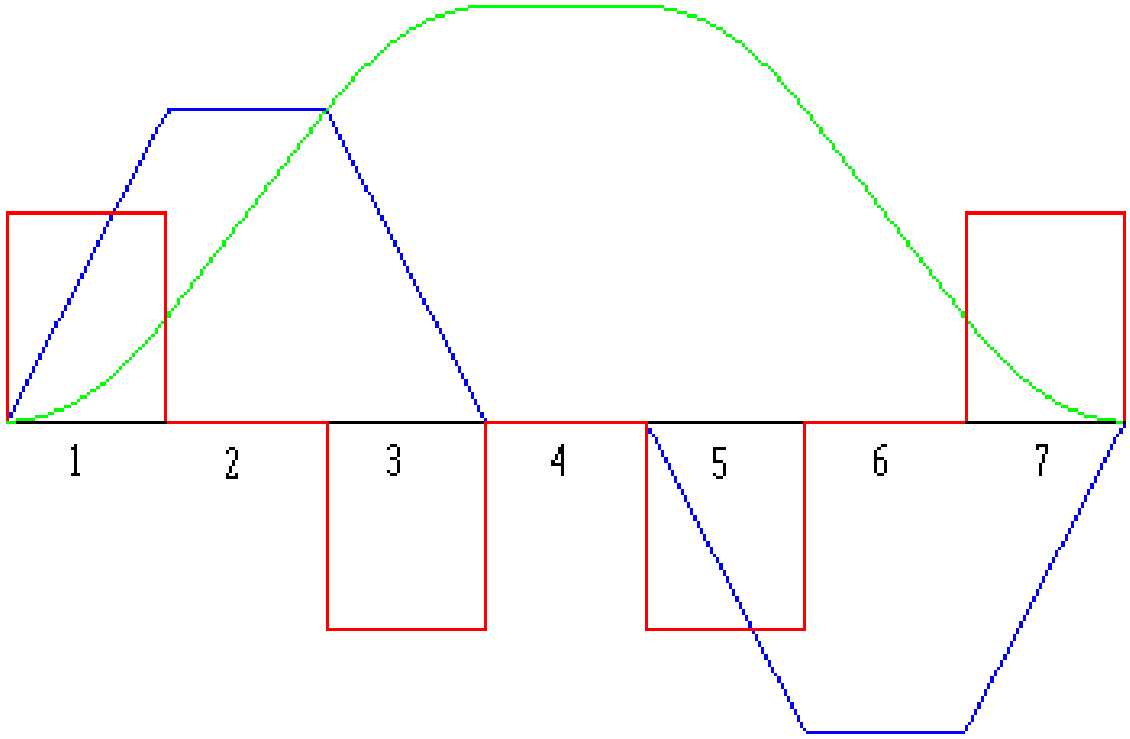


Motion Profiling

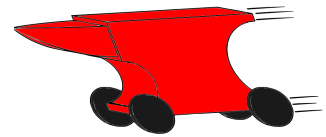




Motion Profiling

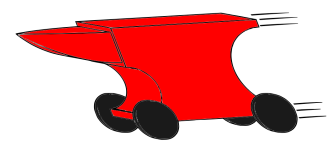


Jerk
Speed
Acceleration



Motion Profiling

```
 MotionProfileGenerator.generateSimpleMotionProfile(
new MotionState(currentPositon, 0, 0, 0),
new MotionState(targetPosition, 0, 0, 0),
MAX_V,
MAX_A,
MAX_J
);
```

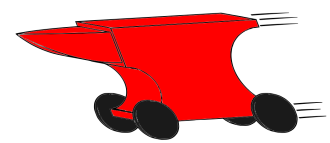


Feedforward Control

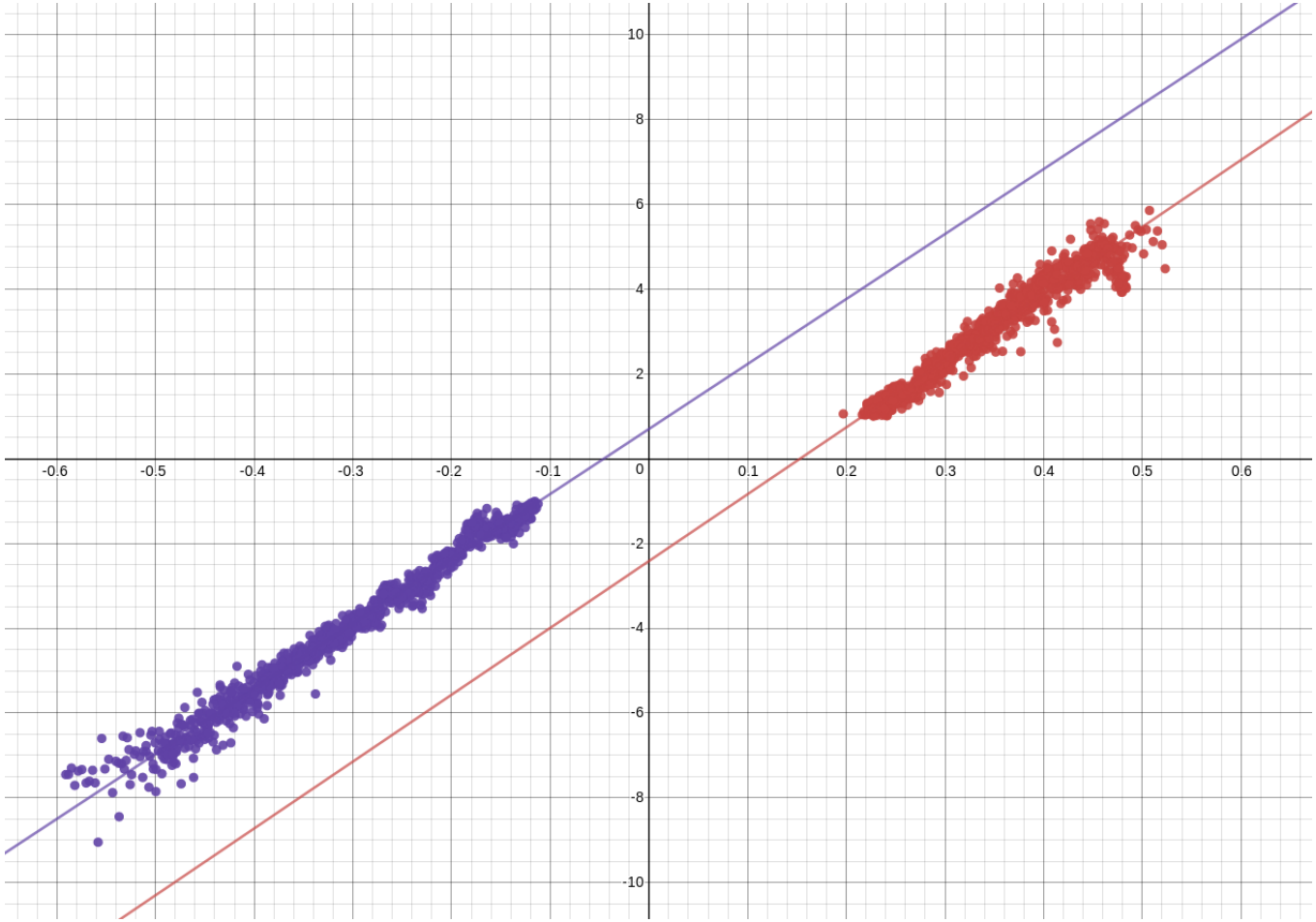
$$voltage \approx k_\omega \omega + k_\tau \tau$$

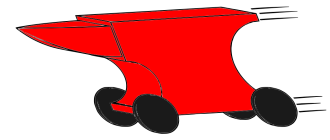
$$v \propto \omega, \; a \propto \tau$$

$$ff(t) = k_v v(t) + k_a a(t) + some\ friction\ stuff$$



Tuning

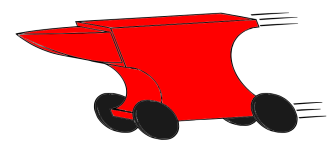




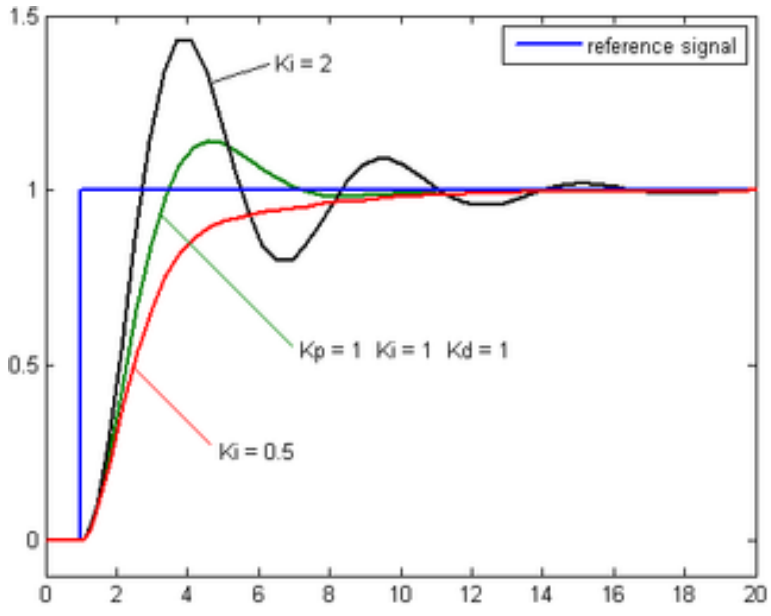
PID Control

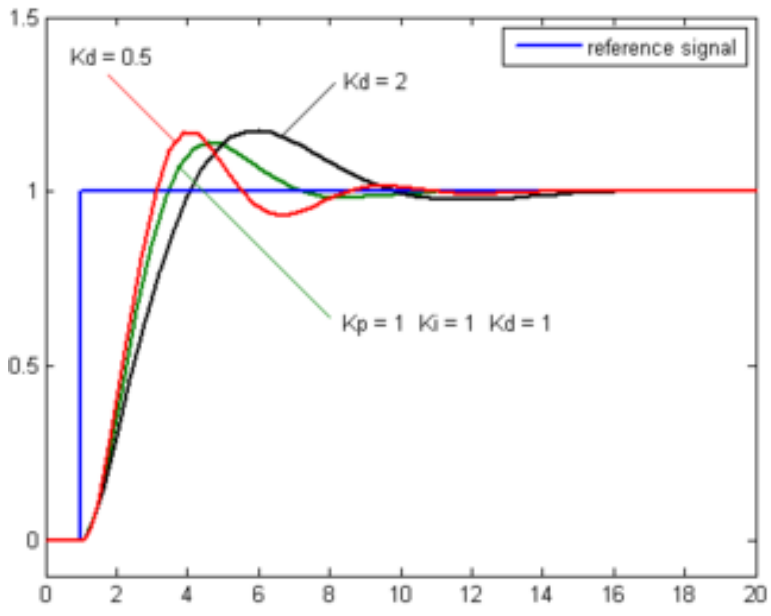
$$c(t) = k_p e(t) + k_i \int_0^t e(\tau) d\tau + k_d \frac{d}{dt} e(t)$$

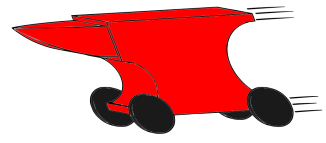
Proportional term: respond to current error
Integral term: respond to constant state error
Derivative term: respond to change in error



PID Control


reference signal
Ki = 2
Kp = 1 Ki = 1 Kd = 1
Ki = 0.5
1.5
1
0.5
0
0 2 4 6 8 10 12 14 16 18 20

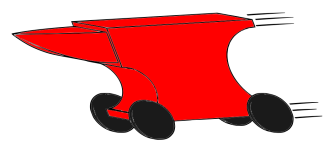

reference signal
Kd = 0.5
Kd = 2
Kp = 1 Ki = 1 Kd = 1
1.5
1
0.5
0
0 2 4 6 8 10 12 14 16 18 20



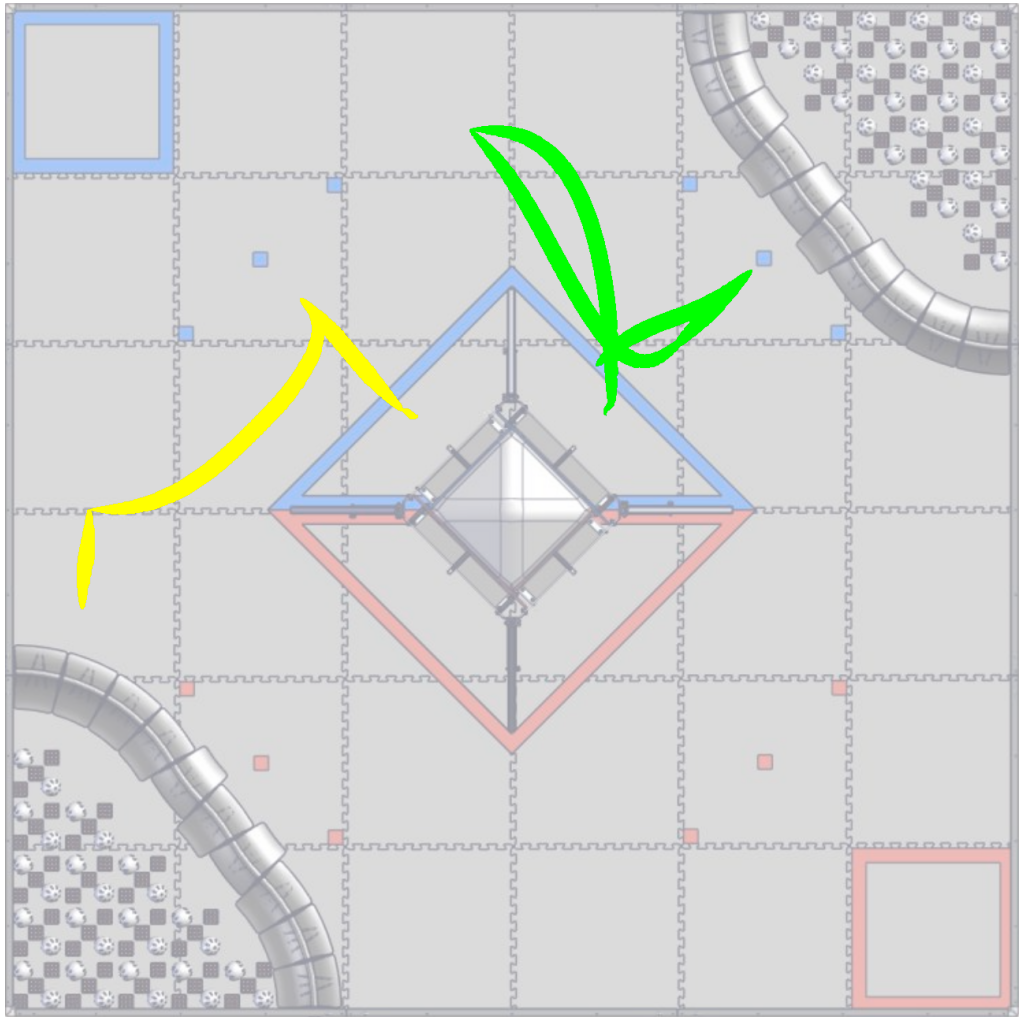
PIDF Controller

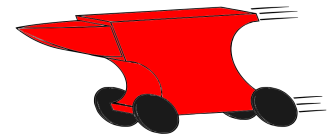
```
 new PIDFController(
new PIDCoefficients(K_P, K_I, K_D),
K_V, K_A, K_STATIC, (x) -> G);
```

```
 controller.setTargetPosition(target);
 controller.update(currentPosition, targetV,
targetA);
```

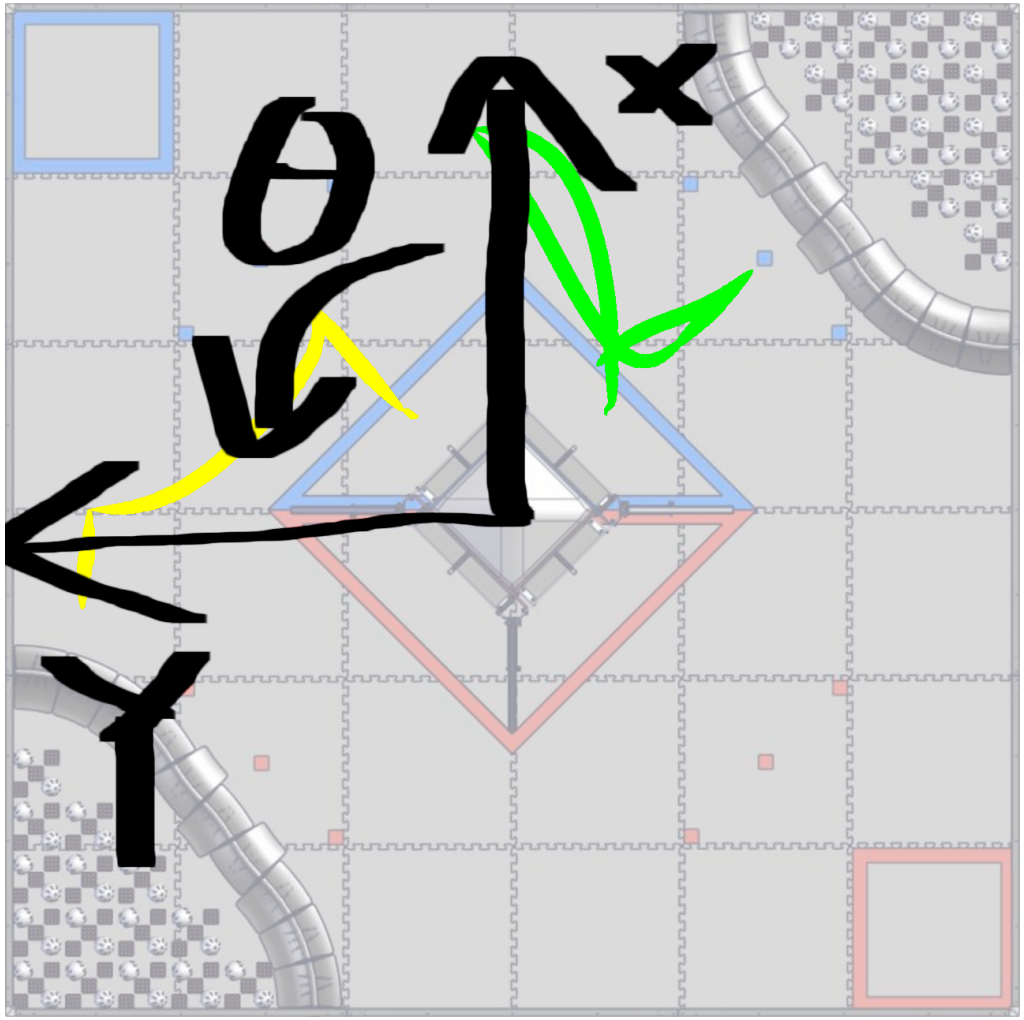


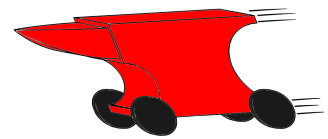
Hermite Splines





Hermite Splines

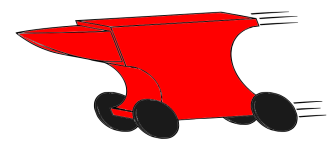




Hermite Splines

$$(x, y) = (p_x(u), p_y(u))$$

$$p(u) = au^5 + bu^4 + cu^3 + du^2 + eu + f$$



Hermite Splines

$$p_x(0) = x_0$$

$$p_x(1) = x_1$$

$$p'_x(0) = cos(\theta_0)$$

$$p'_x(1) = cos(\theta_1)$$

$$p''_x(0) = 0$$

$$p''_x(1) = 0$$

$$p(u) = au^5 + bu^4 + cu^3 + du^2 + eu + f$$