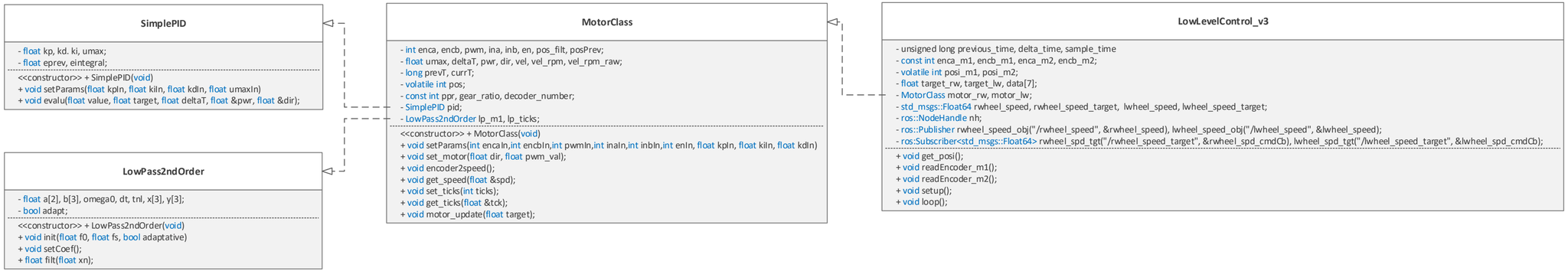
SimplePID
- float kp, kd. ki, umax;
- float eprev, eintegral;
<<constructor>> + SimplePID(void)
+ void setParams(float kpIn, float kiIn, float kdIn, float umaxIn)
+ void evalu(float value, float target, float deltaT, float &pwr, float &dir);
LowPass2ndOrder
- float a[2], b[3], omega0, dt, tnI, x[3], y[3];
- bool adapt;
<<constructor>> + LowPass2ndOrder(void)
+ void init(float f0, float fs, bool adaptative)
+ void setCoef();
+ float filt(float xn);
MotorClass
- int enca, encb, pwm, ina, inb, en, pos_filt, posPrev;
- float umax, deltaT, pwr, dir, vel, vel_rpm, vel_rpm_raw;
- long prevT, currT;
- volatile int pos;
- const int ppr, gear_ratio, decoder_number;
- SimplePID pid;
- LowPass2ndOrder lp_m1, lp_ticks;
<<constructor>> + MotorClass(void)
+ void setParams(int encaIn,int encbIn,int pwmIn,int inaIn,int inbIn,int enIn, float kpIn, float kiIn, float kdIn)
+ void set_motor(float dir, float pwm_val);
+ void encoder2speed();
+ void get_speed(float &spd);
+ void set_ticks(int ticks);
+ void get_ticks(float &tck);
+ void motor_update(float target);
LowLevelControl_v3
- unsigned long previous_time, delta_time, sample_time
- const int enca_m1, encb_m1, enca_m2, encb_m2;
- volatile int posi_m1, posi_m2;
- float target_rw, target_lw, data[7];
- MotorClass motor_rw, motor_lw;
- std_msgs::Float64 rwheel_speed, rwheel_speed_target, lwheel_speed, lwheel_speed_target;
- ros::NodeHandle nh;
- ros::Publisher rwheel_speed_obj("/rwheel_speed", &rwheel_speed), lwheel_speed_obj("/lwheel_speed", &lwheel_speed);
- ros:Subscriber<std_msgs::Float64> rwheel_spd_tgt("/rwheel_speed_target", &rwheel_spd_cmdCb), lwheel_spd_tgt("/lwheel_speed_target", &lwheel_spd_cmdCb);
+ void get_posi();
+ void readEncoder_m1();
+ void readEncoder_m2();
+ void setup();
+ void loop();