ACC 구현

ToF 센서 기반 거리 측정

- 100ms 주기로 can-callback
- 3바이트 조합하여 ToF 센싱 값을 mm 거리값으로 변환(uint_32t)
- 변환한 속도 값을 토픽(/tof_distance)으로 발행

```
/* callback function */
void can_callback(const can_msgs::msg::Frame::SharedPtr msg) {

    // tof can id = 0x200
    if (msg->id == 0x200) {
        unsigned int tofVal = (msg->data[2] << 16) | (msg->data[1] << 8) | msg->data[0]; // 3바이트를 조합해서 센싱 값을 mm 값으로 변환

        auto tof_msg = std_msgs::msg::UInt32();
        tof_msg.data = tofVal;

        tof_pub_->publish(tof_msg); // topic /tof_distance_mm로 발행
    }
}
```

Adaptive Cruise Control (ACC)

- ToF 센서 값(토픽)이 발행될 때마다 call_back
- ACC 노드는 모드 관리 노드의 토픽(현재 노드)를 구독
  → 전체 시스템의 현재 모드를 가져옴
- 현재 노드 == CRUISE인 경우에만 동작

```
/* callback function 2 */
void tof_callback(const std_msgs::msg::UInt32::SharedPtr msg){

    // 현재 저장된 모드가 Mode::CRUISE 아닐 경우 skip (즉시 종료)
    if (current_mode_ != Mode::CRUISE){
        return;
    }
```

ACC - 긴급 제동

- current_distance < braking_dist 인 경우
- braking_dist = (braking_factor * cruise_speed)
  + panic_dist
- PI 오차 누적값을 0으로 초기화

- 속도 = 0

```
// case 1: predictive braking (AEB)
if (current_distance < braking_dist_) {
    speed_new = 0.0f;
    error_integral_ = 0.0f;

    RCLCPP_WARN_ONCE(this->get_logger(), "PREDICTIVE BRAKING! Dist: %.1f < Needed: %.1f",
        current_distance, braking_dist_);
}
```

ACC - 정속 주행

- current_distance > detect_threshold인 경우
- detect_threshold = 목표거리 * 1.5
- PI 오차 누적값을 0으로 초기화
- 속도 = cruise_speed

```
// case 2: cruise
else if (current_distance > detect_threshold_) {
    speed_new = cruise_speed_;
    error_integral_ = 0.0f;
}
```

ACC - PI 제어

- braking_dist < current_dist < detect_threshold인 경우
- 오차 누적값 += 거리 오차 * 0.1
  (거리값은 mm 단위, 속도는 0.0~1.0이기 때문)
- 오차 누적값이 최대값을 넘지 않도록 처리 (**Anti-Windup**)
- **속도 = (Kp * 현재 오차) + (Ki * 누적 오차)**

```
// case 3: PI control
else {
    float error_dist = current_distance - target_dist_;
    error_integral_ += error_dist * DT;
    // anti-windup
    error_integral_ = std::max(std::min(error_integral_, MAX_INTEGRAL), -MAX_INTEGRAL);

    speed_new = kp_ * error_dist + ki_ * error_integral_;
}
```

ACC - 안전 장치

- 속도를 토픽(/planner/speed)으로 발행하기 전
- 결정된 속도가 음수 혹은 최대값(cruise_speed) 을 넘는 경우가 없도록 처리

```
// post processing for safty
speed_new = std::max(0.0f, std::min(speed_new, static_cast<float>(cruise_speed_)));
```

ACC - Creep

- 현재 모드가 CRUISE가 아닌 경우
  (DODGE, SWITCH, TURN)
- 속도 = creep_speed_output (일정한 속도 값을 토픽으로 발행)

```
/* callback function 2 */
void timer_callback(){
    // 현재 저장된 모드가 normal일 경우 즉시 종료 (skip)
    if (current_mode_ == Mode::MANUAL || current_mode_ == Mode::CRUISE) {
        return;
    }

    const float creep_speed_output = 0.15f;
```