```
            car.x = j[1]["x"];
            car.y = j[1]["y"];
            car.s = j[1]["s"];
            car.d = j[1]["d"];
            car.yaw = j[1]["yaw"];
            car.speed = j[1]["speed"];

            Path previous_path;

            auto previous_path_x = j[1]["previous_path_x"];
            auto previous_path_y = j[1]["previous_path_y"];

            int prev_size = previous_path_x.size();
//          printf("Previous size: %d\n", prev_size);
            for (int i = 0; i < prev_size; ++i){
              previous_path.x.push_back(previous_path_x[i]);
              previous_path.y.push_back(previous_path_y[i]);
            }

            EndPath end_path;

            end_path.s = j[1]["end_path_s"];
            end_path.d = j[1]["end_path_d"];

            auto sensor_fusion = j[1]["sensor_fusion"];
            int sensor_fusion_size = sensor_fusion.size();
//          printf("sensor_fusion_size: %d\n", sensor_fusion_size);
            vector<vector<double>> sen_fusion;
            for (int i = 0; i < sensor_fusion_size; ++i){
              sen_fusion.push_back(sensor_fusion[i]);
            }

            path_planner->set_sensor_fusion(sen_fusion);
            path_planner->set_previous_path(previous_path);
            path_planner->set_car(car);
            path_planner->set_end_path(end_path);
            path_planner->set_ref_vel(49.5); //mph
```

AWESOME

Brilliant work writing a separate class and functions to perform the path planning for the ego vehicle. It ma

```

            path_planner->calc_next_xy_vals();

            json msgJson;
            msgJson["next_x"] = path_planner->get_next_x_vals();
            msgJson["next_y"] = path_planner->get_next_y_vals();

            auto msg = "42[\"control\","+ msgJson.dump()+"]";

            //this_thread::sleep_for(chrono::milliseconds(1000));
            ws.send(msg.data(), msg.length(), uWS::OpCode::TEXT);

          }
        } else {
          // Manual driving
          std::string msg = "42[\"manual\",{}]";
          ws.send(msg.data(), msg.length(), uWS::OpCode::TEXT);
```