NMPC 记录

source gp_mpc_venv/bin/activate

cd catkin_ws/

catkin build

source devel/setup.bash

export PYTHONPATH=$PYTHONPATH:/home/zyb/workspace/catkin_ws/src/data_driven_mpc/ros_gp_mpc
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:"/home/zyb/workspace/acados/lib"

export ACADOS_SOURCE_DIR="/home/zyb/workspace/acados"

roscd ros_gp_mpc

python src/experiments/trajectory_test.py

运行该框架

2024.01.23

在trajectory_test.py同样的位置写一个我们能用的mpc，my_nmpc_test.py

MPC参数：

函数套娃：prepare_quadrotor_mpc -> Quad3DMPC （元素包含一个Quad3DOptimizer），prepare_quadrotor_mpc返回一个Quad3DMPC类对象

```
quad_mpc = Quad3DMPC(my_quad, t_horizon=t_horizon, optimization_dt=node_dt, simulation_dt=simulation_dt,
                     q_cost=q_diagonal, r_cost=r_diagonal, n_nodes=n_mpc_nodes,
                     pre_trained_models=pre_trained_models, model_name=quad_name, q_mask=q_mask, rdrv_d_mat=rdrv_d)
Quad3DOptimizer(my_quad, t_horizon=t_horizon, n_nodes=n_nodes,
                        q_cost=q_cost, r_cost=r_cost,
                        B_x=self.B_x, gp_regressors=self.gp_ensemble,
                        model_name=model_name, q_mask=q_mask,
                        solver_options=solver_options, rdrv_d_mat=rdrv_d_mat)
```

quad_3d.py 中Quad3DMPC类修改UAV参数（龙塔4th可参考）

quad_3d_optimizer.py中Quad3DOptimizer类修改MPC参数

主函数

my_quad = Quadrotor3D(**simulation_options)机身参数设置，看是否考虑更多更精细的情况

t_horizon = 1s

n_mpc_nodes = 10个

node_dt = t_horizon / n_mpc_nodes=0.1s

simulation_dt = 5e-4 (the smaller the more "continuous"-like simulation)仿真相关，不知是否有用

q_cost=q_diagonal

r_cost=r_diagonal

q_mask=q_mask

pre_trained_models=pre_trained_models = None

rdrv_d_mat=rdrv_d = None

B_x空矩阵

solver_options=None，留下了更换solver的接口

set_reference_state() 不知是否还要设置什么

w_opt, x_pred = quad_mpc.optimize(use_model=model_ind, return_x=True)

Quad3DMPC类（自己写的仿真器）

Quad3DMPC..quad.update(ref_u, self.simulation_dt)动力学模型

似乎并未借助casadi库?

```
  def optimize(self, use_model=0, return_x=False):



    quad_current_state = self.quad.get_state(quaternion=True, stacked=True)
    quad_gp_state = self.quad.get_gp_state(quaternion=True, stacked=True)

    # Remove rate state for simplified model NLP
    out_out = self.quad_opt.run_optimization(quad_current_state,
use_model=use_model, return_x=return_x,
                                             gp_regression_state=quad_gp_state)
    return out_out
```

Quad3DOptimizer类（2024.01.24）

动力学等式

$$\dot{\boldsymbol{x}} = \begin{bmatrix} \dot{\boldsymbol{p}}_{WB} \\ \dot{\boldsymbol{q}}_{WB} \\ \dot{\boldsymbol{v}}_{WB} \\ \dot{\boldsymbol{\omega}}_{B} \end{bmatrix} = \boldsymbol{f}_{dyn}(\boldsymbol{x}, \boldsymbol{u}) = \begin{bmatrix} \boldsymbol{v}_W \\ \boldsymbol{q}_{WB} \cdot \begin{bmatrix} 0 \\ \boldsymbol{\omega_B}/2 \end{bmatrix} \\ \boldsymbol{q}_{WB} \odot \boldsymbol{T}_B + \boldsymbol{g}_W \\ \boldsymbol{J}^{-1}\left(\boldsymbol{\tau}_B - \boldsymbol{\omega}_B \times \boldsymbol{J}\boldsymbol{\omega}_B\right) \end{bmatrix}$$

self.quad_xdot_nominal = self.quad_dynamics(rdrv_d_mat)

```
  def quad_dynamics(self, rdrv_d):
```

```
    x_dot = cs.vertcat(self.p_dynamics(), self.q_dynamics(),
self.v_dynamics(rdrv_d), self.w_dynamics())
    return cs.Function('x_dot', [self.x[:13], self.u], [x_dot], ['x', 'u'], ['x_dot']) anyway
最后返回quad_xdot_nominal也是个函数,其输入为['x', 'u']输出为['x_dot']
  def p_dynamics(self):
    return self.v
  def q_dynamics(self):
    return 1 / 2 * cs.mtimes(skew_symmetric(self.r), self.q)
  def v_dynamics(self, rdrv_d):
    f_thrust = self.u * self.quad.max_thrust
    g = cs.vertcat(0.0, 0.0, 9.81)
    a_thrust = cs.vertcat(0.0, 0.0, f_thrust[0] + f_thrust[1] + f_thrust[2] + f_thrust[3])
/ self.quad.mass

    v_dynamics = v_dot_q(a_thrust, self.q) - g

    if rdrv_d is not None:
      # Velocity in body frame:
      v_b = v_dot_q(self.v, quaternion_inverse(self.q))
      rdrv_drag = v_dot_q(cs.mtimes(rdrv_d, v_b), self.q)
      v_dynamics += rdrv_drag

    return v_dynamics

def w_dynamics(self):
    f_thrust = self.u * self.quad.max_thrust

    y_f = cs.MX(self.quad.y_f)
    x_f = cs.MX(self.quad.x_f)
    c_f = cs.MX(self.quad.z_l_tau)
    return cs.vertcat(
      (cs.mtimes(f_thrust.T, y_f) + (self.quad.J[1] - self.quad.J[2]) * self.r[1] *
self.r[2]) / self.quad.J[0],
      (-cs.mtimes(f_thrust.T, x_f) + (self.quad.J[2] - self.quad.J[0]) * self.r[2] *
self.r[0]) / self.quad.J[1],
      (cs.mtimes(f_thrust.T, c_f) + (self.quad.J[0] - self.quad.J[1]) * self.r[0] *
self.r[1]) / self.quad.J[2])

acados_models, nominal_with_gp = self.acados_setup_model(
      self.quad_xdot_nominal(x=self.x, u=self.u)['x_dot'], model_name)
```

函数：Builds an Acados symbolic models using CasADi expressions.

Return acados_models, dynamics_equations 我们使用时返回的是只有一个元素的字典acados_models[0]，dynamics_equations[0]，model数据类型是这样来的AcadosModel()，dynamics_equations仍是casadi.Fuction

2024.01.25 ocp = AcadosOcp() 创建OCP问题（初始化）

```
#nx:状态数，nu:输入数，ny:x与u之和
nx = key_model.x.size()[0]
nu = key_model.u.size()[0]
ny = nx + nu
# 额外参数，暂时没有用到
n_param = key_model.p.size()[0] if isinstance(key_model.p, cs.MX)
else 0
#设置环境变量后获得系统中ACADOS的安装路径
acados_source_path = os.environ['ACADOS_SOURCE_DIR']
sys.path.insert(0, '../common')#sys.path是一个列表，包含了Python解释器
查找模块和包时会搜索的所有目录

# 构建OCP
ocp = AcadosOcp()
## 设置ACADOS系统引用以及库的路径（因为ACADOS最后将以C的形式运行，所以必须
设置正确）
ocp.acados_include_path = acados_source_path + '/include'
ocp.acados_lib_path = acados_source_path + '/lib'
## 设置模型
ocp.model = key_model
ocp.dims.N = self.N
ocp.solver_options.tf = t_horizon

# Initialize parameters
ocp.dims.np = n_param
ocp.parameter_values = np.zeros(n_param)

#代价函数
ocp.cost.cost_type = 'LINEAR_LS'
ocp.cost.cost_type_e = 'LINEAR_LS'

ocp.cost.W = np.diag(np.concatenate((q_diagonal, r_cost)))
ocp.cost.W_e = np.diag(q_diagonal)
terminal_cost = 0 if solver_options is None or not
solver_options["terminal_cost"] else 1
ocp.cost.W_e *= terminal_cost

ocp.cost.Vx = np.zeros((ny, nx))
ocp.cost.Vx[:nx, :nx] = np.eye(nx)
ocp.cost.Vu = np.zeros((ny, nu))
ocp.cost.Vu[-4:, -4:] = np.eye(nu)

ocp.cost.Vx_e = np.eye(nx)

# Initial reference trajectory (will be overwritten)
#随便初始化一下，维数对就行
x_ref = np.zeros(nx)
ocp.cost.yref = np.concatenate((x_ref, np.array([0.0, 0.0, 0.0,
0.0])))
ocp.cost.yref_e = x_ref

# Initial state (will be overwritten)
```

```
#随便初始化一下，维数对就行
ocp.constraints.x0 = x_ref

# Set constraints（现在只有性能约束，还需要安全约束）
ocp.constraints.lbu = np.array([self.min_u] * 4)
ocp.constraints.ubu = np.array([self.max_u] * 4)
ocp.constraints.idxbu = np.array([0, 1, 2, 3])

# Solver options
ocp.solver_options.qp_solver = 'FULL_CONDENSING_HPIPM'
ocp.solver_options.hessian_approx = 'GAUSS_NEWTON'
ocp.solver_options.integrator_type = 'ERK'
ocp.solver_options.print_level = 0
ocp.solver_options.nlp_solver_type = 'SQP_RTI' if solver_options is
None else solver_options["solver_type"]

# Compile acados OCP solver if necessary
#生成配置文件以及相应的C代码，ACADOS可以将仿真器也自动生成（我们已经有了
quad_3d类了不需要仿真器）
json_file = os.path.join(self.acados_models_dir, key_model.name +
'_acados_ocp.json')
self.acados_ocp_solver[key] = AcadosOcpSolver(ocp,
json_file=json_file)
```

2024.01.26 quad_mpc.optimize()函数中的 Quad3DOptimizer.run_optimization()

（初始化之后运行时需要设置的参数）

Replan过程中需要更新的输入
当前状态
目标位置和raw角
点云转化来的每一时间步都要加的安全约束
初始状态（当前状态）的约束?

gp_regressors=None->gp_reg_ensemble = None->with_gp=False

导致

set_reference_state()函数返回值为0，即model_ind=0

那么运行下面代码时最终用的就是0号求解器（虽然本来也就一个）

```
#主函数调用
w_opt, x_pred = quad_mpc.optimize(use_model=model_ind,
return_x=True)
```

```
def optimize(self, use_model=0, return_x=False):


    quad_current_state = self.quad.get_state(quaternion=True,
stacked=True)
    quad_gp_state = self.quad.get_gp_state(quaternion=True,
stacked=True)#我们不需要加噪声，quad_gp_state = None
```

```
# Remove rate state for simplified model NLP
    out_out = self.quad_opt.run_optimization(quad_current_state,
use_model=use_model, return_x=return_x,

gp_regression_state=quad_gp_state)

    return out_out
```

```
def run_optimization(self, initial_state=None, use_model=0,
return_x=False, gp_regression_state=None):


    if initial_state is None:
        initial_state = [0, 0, 0] + [1, 0, 0, 0] + [0, 0, 0] + [0,
0, 0]

    # Set initial state. Add gp state if needed
    x_init = initial_state
    x_init = np.stack(x_init)

    # Set initial condition, equality constraint
    self.acados_ocp_solver[use_model].set(0, 'lbx', x_init)
    self.acados_ocp_solver[use_model].set(0, 'ubx', x_init)

    # Set parameters（暂时没用）
    if self.with_gp:
        gp_state = gp_regression_state if gp_regression_state is not
 None else initial_state
        self.acados_ocp_solver[use_model].set(0, 'p',
np.array(gp_state + [1]))
        for j in range(1, self.N):
            self.acados_ocp_solver[use_model].set(j, 'p',
np.array([0.0] * (len(gp_state) + 1)))

    # Solve OCP
    self.acados_ocp_solver[use_model].solve()

    # Get u
    w_opt_acados = np.ndarray((self.N, 4))
    x_opt_acados = np.ndarray((self.N + 1, len(x_init)))
    x_opt_acados[0, :] = self.acados_ocp_solver[use_model].get(0,
"x")
    for i in range(self.N):
        w_opt_acados[i, :] =
self.acados_ocp_solver[use_model].get(i, "u")
        x_opt_acados[i + 1, :] =
self.acados_ocp_solver[use_model].get(i + 1, "x")

    w_opt_acados = np.reshape(w_opt_acados, (-1))
    return w_opt_acados if not return_x else (w_opt_acados,
x_opt_acados)
```

casadi、acados相关问题

- [x] ~~1、self.p = casadi.MX.sym('p', 3)为什么casadi库总要有'p'，感觉有点多余，它的作用是什么，不是已经有返回的对象叫做self.p了？~~

z是什么？

SZ SHEN Zhipeng 2 小时前
这个可以看文档里的最优控制问题 formulation：https://github.com/acados/acados/blob/master/docs/problem_formulation/problem_formulation_ocp_mex.pdf

一博 赵一博 1 小时前
看了一下，不知道我理解的对不对 QAQ

执行print(elf.p)时，会打印出p而不是变量的内存地址或者其他不可读的信息，这在调试复杂的数学表达式时，能够容易理解每个符号变量的含义；这个名称在CasaADI的Function对象中标识输入和输出变量，比如f = Fuction('f', [self.p], [self.p**2]),调用f时，可以f(p=2)，返回值是一个字典，quad_xdot_nominal(x=self.x, u=self.u)['x_dot']中['x_dot']提取字典中键为x_dot的元素

- [ ] 2、显式动力学和隐式动力学（不影响使用）

```python
def fill_in_acados_model(x, u, p, dynamics, name):

    x_dot = cs.MX.sym('x_dot', dynamics.shape)
    f_impl = x_dot - dynamics

    # Dynamics model
    model = AcadosModel()
    model.f_expl_expr = dynamics#显式动力学
    model.f_impl_expr = f_impl#隐式动力学
    model.x = x
    model.xdot = x_dot
    model.u = u
    model.p = p
    model.name = name

    return model
```

- [x] ~~3、ocp.cost.V，acados中V类矩阵作用？~~

- [x] ~~4、z是什么？~~

https://docs.acados.org/python_interface/index.html?highlight=ocp%20cost#module-acados_template.acados_ocp_cost

In case of LINEAR_LS: stage cost is $l(x,u,z) = ||V_x\,x + V_u\,u + V_z\,z - y_{\text{ref}}||^2_W$, terminal cost is $m(x) = ||V_x^e\,x - y_{\text{ref}}^e||^2_{W^e}$

我们W=0

$$\min_{x(\cdot),u(\cdot),z(\cdot),s(\cdot),s^e} \int_0^T l(x(\tau),u(\tau),z(\tau),p) + \frac{1}{2}\begin{bmatrix} s_l(\tau) \\ s_u(\tau) \\ 1 \end{bmatrix}^\top \begin{bmatrix} Z_l & 0 & z_l \\ 0 & Z_u & z_u \\ z_l^\top & z_u^\top & 0 \end{bmatrix} \begin{bmatrix} s_l(\tau) \\ s_u(\tau) \\ 1 \end{bmatrix} \mathrm{d}\tau + m(x(T),z(T),p) + \frac{1}{2}\begin{bmatrix} s_l^e \\ s_u^e \\ 1 \end{bmatrix}^\top \begin{bmatrix} Z_l^e & 0 & z_l^e \\ 0 & Z_u^e & z_u^e \\ z_l^{e\top} & z_u^{e\top} & 0 \end{bmatrix} \begin{bmatrix} s_l^e \\ s_u^e \\ 1 \end{bmatrix} \tag{1}$$

- algebraic state vector $z : \mathbb{R} \to \mathbb{R}^{n_z}$

Z是代数状态，例如，在一个车辆路径规划问题中，我们可能有状态向量x（包括车辆的位置和速度），控制向量u（包括车辆的加速度和转向角），以及代数状态z（包括车辆的航向角或者其他的路径参数）。在这种情况下，代价函数可能会包含x，u和z的项。

- [x] ~~5、如何添加每一步的约束？~~

使用AcadosOcpSolver.set函数，解析如下

set(*stage_: int, field_: str, value_*)

Set numerical data inside the solver.

param stage	integer corresponding to shooting node
param field	string in ['x', 'u', 'pi', 'lam', 't', 'p', 'xdot_guess', 'z_guess']

每一步都要加入安全约束，并实时更新

大部分AcadosOcpSolver类的函数结构都类似