Project Report

Course: Convex Optimization

Xiao HU

Kongens Lyngby
June 21, 2018

Abstract

This report will present a autonomous trajectory generation method using semidefinite programming, which includes the formulation of the original problem, minimum jerk based trajectory generation as well as time re-allocation using semidefinite relaxation and semidefinite programming. The task is done as a final project for course "Convex Optimization" at DTU.

Contents

Chapter 1

Contents

1.1 Problem Formulation

According to [5], for any robots with differentiable flat property, their trajectory can be represented using piece-wise polynomial as

$$f(t) = p_0 + tp_1 + t^2p_2 + t^3p_3 + t^4p_4 + t^5p_5 \tag{1.1}$$
$$f^{(1)}(t) = p_1 + 2tp_2 + 3t^2p_3 + 4t^3p_4 + 5t^4p_5 \tag{1.2}$$
$$f^{(2)}(t) = 2p_2 + 6tp_3 + 12t^2p_4 + 20t^3p_5 \tag{1.3}$$
$$f^{(3)}(t) = 6p_3 + 24tp_4 + 60t^2p_5 \tag{1.4}$$
$$\tag{1.5}$$

where p_i denoting the i^{th} coefficient of a given polynomial, t being a scalar representing the time.

1.1.1 Objective Function

The cost function or objective function in convex optimization is defined with the integration of the square of k^{th} derivative of $f(t)$:

$$\mathcal{J} = \int_0^T \left(||f^{(k)}(t)||_2^2 \right) dt \tag{1.6}$$

It is clearly to see with known T, Eq. 1.6 can be further written as

$$\mathcal{J} = \mathbf{p}^T Q(T) \mathbf{p} \tag{1.7}$$

where $\mathbf{p}$ being a vector of coefficients of a single polynomial. The extension of Eq. **??** for M piece-wise polynomial segments can be easily written by concatenating their cost matrices in a block-diagonal fashion:

$$\mathcal{J} = \begin{bmatrix} \mathbf{p_1} \\ \vdots \\ \mathbf{p_2} \end{bmatrix}^T \begin{bmatrix} Q_1(T_1) & & \\ & \ddots & \\ & & Q_M T_M \end{bmatrix} \begin{bmatrix} \mathbf{p_1} \\ \vdots \\ \mathbf{p_2} \end{bmatrix} \tag{1.8}$$

1.1.2 Constraints

Normally, constraints for trajectory generation include

- reach a given waypoint, $f(t) = pos$;
- reach given velocities or accelerations at a certain waypoint, $f^{(1)}(t) = vel$, $f^{(2)}(t) = acc$;
- continuity between two adjoint segments to make sure the velocity and acceleration smooth, $f_1^{(1)}(t) = f_2^{(1)}(t)$;
- maximum velocity and maximum acceleration constraint $f^{(1)}(t) \leq v_{max}$, $f^{(2)}(t) \leq a_{max}$;

All of those constraints could be written as $A\mathbf{p} = \mathbf{b}$ or $A\mathbf{p} \leq \mathbf{b}$.

1.1.3 Optimization Problem

Summarizing constraints and objective function together, trajectory generation is nothing more than the following optimization problem:

$$\text{minimize: } \mathcal{J} \tag{1.9}$$
$$\text{subject to: } A_i\mathbf{p_i} = b_i, \text{ for i} = 1 \cdots M \tag{1.10}$$
$$A_i\mathbf{p_i} \leq b_j, \text{ for j} = 1 \cdots M \tag{1.11}$$
$$\tag{1.12}$$

With known t, the objective function is a quadratic form of $\mathbf{p}$. The Hessian matrix Q is the integration of a matrix $\hat{Q} = \mathbf{a}\mathbf{a}^T,\ \mathbf{a} = [0\ 0\ 0\ 6\ 24t\ 60t^2]^T,\ t \geq 0$ (e.g. assuming taking the 3^{rd} derivative), so Q is positive semidefinite. The objective function is convex and all constraints are affine function of $\mathbf{p}$, it can be easily obtained that the trajectory generation problem is nothing more than a QP problem. More specifically, if there are only equality constraints, [6] proposed a way to reformulate as a unconstrained problem. However, I found that for this problem, the KKT matrix is always invertible. As a result, the most straight forward way to solve the KKT matrix and get the primal and dual optimal solutions [1]. It is noted that the KKT matrix in this problem is block diagonal and sparse, so solving a linear equation $Ax = b$ can be efficiently done. The most common way to do trajectory generation is then summarized as: 1. time allocation to choose t; 2. formulate QP problem; 3. solve with QP solver. However, a key problem for this method is how to choose a proper t beforehand. Unproper time allocation or unappropriate t would make the trajectory infeasible. However, if t is an optimized variable, then the optimization problem is non linear and nonconvex.

1.1.4 Semidefinite Relaxation

Now consider a linear mapping function from real time t_{real} to virual time $t_{virtual}$,

$$t_{virtual} = st_{real} \tag{1.13}$$

where s is a scalar. Then a feasible way for trajectory generation would be: trajectory generation using QP with virtual time allocation, finding $\mathbf{s}$ that map from real time to virtual time and fullfill constraints. The objective function for $\mathbf{s}$ would be:

$$\text{minimize: } \sum_{i=1}^{r}(s_i - 1)^2 \tag{1.14}$$

Equality constraints have to be relaxed to inequality constraints:

$$\text{subject to: } poly(s_i, s_i^2, \cdots, s_i^k) \leq b \tag{1.15}$$

Then inspired from [4] and [1], by introducing a new variable $X = \mathbf{x}\mathbf{x}^T$, $\mathbf{x} = [s_i \ s_i^2 \ \cdots]$, The optimization problem can be reformulated using X as a semidefinite programming:

$$\begin{aligned} \text{minimize: } & tr(C_0 X) && (1.16)\\ \text{subject to: } & tr(A_i X) \leq b_i && (1.17)\\ & X \succeq 0 && (1.18)\\ & rank(X) = 1. && (1.19) \end{aligned}$$

The rank constraint is nonconvex, so drop it to get a relaxed problem:

$$\begin{aligned} \text{minimize: } & tr(C_0 X) && (1.20)\\ \text{subject to: } & tr(A_i X) \leq b_i && (1.21)\\ & X \succeq 0 && (1.22)\\ & && (1.23) \end{aligned}$$

Then we can solve it with CVX optimization software [2]. However, there is no free lunch for this relaxation. The key problem here is how to reconstruct x^* from X^*. This will drop into two cases: 1. if$rank(X^*) = 1$, then x^* can easily obtained; 2. if $rank(X^*) > 1$, the optimum value provides a lower bound for the original problem. In this case, there are several ways to obtain a feasible $\tilde{x}$ for the original problem. One way is by eigen decomposition of X^*

$$X* = \sum_{i=1}^{r} \lambda_i \mathbf{q_i} \mathbf{q_i^T} \tag{1.24}$$

where $\lambda_1 \geq \lambda_2 \geq \cdots \geq 0$. The best rank-1 approximation of X^* is $\lambda_1 \mathbf{q_1} \mathbf{q_1}^T$. Then $\tilde{x} = \sqrt{\lambda_1}\mathbf{q_1}$ can be defined as a candidate solution provided $\tilde{x}$ is feasible. A more complicated method based on randomization can be applied, which shows a better performance in [4].

1.2 Implementation

The implementation was done in *MatLab* with the help of sevreal convex optimization toolbox including CVX[1] [2], GloptiPoly[2] [3]. The implementation can be split into 3 parts:

[1] `http://cvxr.com/cvx/`
[2] `http://homepages.laas.fr/henrion/software/gloptipoly/`

Figure 1.1: *MatLab GUI program and one trial example.*

- trajectory generaion using QP with initial time allocation.
- polynomial evaluation and extreme points extraction. Since the real constraints I would like to add is $f^{(1)} \geq b,\ for\ t \in [T_{i-1},\ T_i]$, the constraint is satisfied if and only if all the constraint is met on every extreme points. Because the trajectory is represented with polynomial, its derivatives are all polynomials and their roots can be easily found (actually analytical solution for polynomial less than 5^{th} order).
- time reallocation using semidefinite relaxation and semidefinite programming. A side effect of adding relaxation is that equality constraints have to be relaxed as inequality constraints. Consequently, some waypts can not be visited and there will be discontinuities between adjoint segments. A trade off solution is to do a local refinement using QP.

A *MatLab* GUI program is designed to verify its performance. A snapshot of the program is shown in Fig 1.1. It can be seen from Fig 1.1 that the generated trajectory is very smooth with all velocity and acceleration continuous and all constraints (denoted with red solid lines) satisfied.

1.3 Summary & Future works

In this project, a trajectory generation method using semidefinite relaxation is investigated. By using relaxation, the nonconvex program becomes a standard semidefinite programming problem, which can be solved efficiently.

A potential future work would be optimize t and polynomial coefficients jointly by using semidefinite relaxation. Then for each segment, at least 10 variables need to be defined. A potential problem would be the high dimemsion of optimized variables.

This work is done with a great help from Prof. Martin S. Andersen.

Bibliography

[1] S. Boyd and L. Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[2] M. Grant, S. Boyd, and Y. Ye. Cvx: Matlab software for disciplined convex programming, 2008.

[3] D. Henrion, J.-B. Lasserre, and J. Löfberg. Gloptipoly 3: moments, optimization and semidefinite programming. *Optimization Methods & Software*, 24(4-5):761–779, 2009.

[4] Z.-Q. Luo, W.-K. Ma, A. M.-C. So, Y. Ye, and S. Zhang. Semidefinite relaxation of quadratic optimization problems. *IEEE Signal Processing Magazine*, 27(3):20–34, 2010.

[5] D. Mellinger and V. Kumar. Minimum snap trajectory generation and control for quadrotors. In *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, pages 2520–2525. IEEE, 2011.

[6] C. Richter, A. Bry, and N. Roy. Polynomial trajectory planning for aggressive quadrotor flight in dense indoor environments. In *Robotics Research*, pages 649–666. Springer, 2016.