

## **Exercise 1: OLCAR**

#### $LQR \rightarrow ILQC$



Alexander W. Winkler | 17.03.2015 | 1

## **General Information**

- Download from <u>http://www.adrl.ethz.ch/doku.php/adrl:education:lecture:fs2015</u>
- Code/Answers submitted by Wed, 15.4 through website
  - Code (only \*\_Design.m files)
  - Pdf (max. 1 page): 1-3 sentences per question
- Interviews on Fr, 17.4
  - Group signup for timeslots will be available on course website
  - 10min interview/group
  - Graded pass/fail
  - Grade boost: Ex.1: +0.1, Ex. 2: +0.05, Ex 3. +0.1
- Office hours as usual for questions



## Introduction

- Goal of exercises 1 & 3: design a controller for a quadrotor
  - Exercise 1: Model-based ILQC controller
    - Model accurate and complies with simulation/reality
  - (Exercise 2: Reinforcement learning)
  - Exercise 3: Adapt bias-model based controller
    - Controller model and simulation model differ



## **Quadrotor – AscTec Hummingbird**



#### Technical Data – AscTec Hummingbird

UAV Type	Quadcopter			
Onboard computer	Up to Intel® Atom™ Processor Z530			
Size	540 x 540 x 85,5 mm			
Max. take off weight	0,71 kg			
Max. payload	200 g			
Flight time incl. payload	20 min.			
Range	4,500 m ASL, 1,000 m AGL			
Max. airspeed	15 m/s			
Max. climb rate	5 m/s			
Max. thrust	20 N			

http://www.asctec.de/en/uav-uas-drone-products/asctec-hummingbird/#pane-0-1



## **Quadcopter Model**

- State  $\mathbf{x} = [x, y, z, \phi, \theta, \psi, \dot{x}, \dot{y}, \dot{z}, \dot{\phi}, \dot{\theta}, \dot{\psi}]^T \in \mathbb{R}^{12}$
- Input  $\mathbf{u} = [F_z, M_x, M_y, M_z]^T \in \mathbb{R}^4$

- System dynamics:
  - underactuated
  - non-linear [Model.f\_mode1(x,u)]

$$\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, \mathbf{u}) = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}$$



## **Exercise 1: Move quadrotor to goal state**

- LQR
- LQR with via point
- ILQC
- ILQC with via point



#### ETHzürich

- main\_ex1.m:
  - Task\_Design
  - Cost\_Design
  - LQR\_Design

- ILQC\_Design
- (Visualization)
- 📝 Editor /home/winklera/git/MATLAB/l olcar 2015 ex1 code/main ex1.m main ex1.m x ILOC Design.m x LOR Design.m x Cost Design.m x Task Design.m x + %% OLCAR - Exercise 1 - TLOC 1 2 close all: clearvars: clc: 3 4 addpath(genpath(pwd)): % adds folders and subfolders to path 5 % To generate plots of LQR/ILQG rollout 6 plotvec = {'guad pos noLoad'.'guad angles noLoad'.'control input noLoad'.'rotor thrust noLoad'}; 7 create pdf = [0 0 0 0]: % for which plots should a pdf be created 8 plot ind = [1 2 3 4 ]: % which data to plot on screen 9 10 11 **%% Task definition** 12 -Task = Task Design(); 13 14 %Load the dynamic model of the guadcopter 15 load('Ouadrotor Model.mat'.'Model'): % save as structure "Model" 16 17 % Define cost function 18 -Task.cost = Cost Design( Model.param.mO. Task ): 19 20 21 22 Se Problem 1: Initial controller design 23 -[Initial Controller. Cost LOR] = LOR Design(Model. Task): 24 25 % Visualization of LOR controller 26 sim out lgr = Quad Simulator(Model,Task,Initial Controller); 27 disp('LQR controller performance:'); 28 fprintf('Cost with LOR controller (metric: LOR cost function!):  $J = ... J_n'$ . Cost LOR): 29 fprintf('Start Quadcopter position: x = %.3f, y = %.3f, z = %.3f \n', sim out lqr.x(1:3,1)); 30 fprintf('Final Ouadcopter position: x = %.3f, y = %.3f, z = %.3f \n\n'. sim out lor.x(1:3.end)): 31 -Visualize(sim out lqr,Model.param,'plot mode',1); 32 % Plot Result(sim out lgr,Model.param,'plots',plotvec(plot ind),'file',create pdf(plot ind),'path',pwd) 33 34 35 %% comment out to proceed to Problem 2 36 return: 37 38 39 **%% Problem 2: ILOC controller design** 40 t cpu ≡ cputime; 41 -[ILQC Controller, Cost] = ILQC Design(Model,Task,Initial Controller,@Quad Simulator); 42 t cpu = cputime - t cpu; 43 44 % Visualization of ILOC controller 45 sim out ilgc = Quad Simulator(Model,Task,ILQC Controller); 46 fprintf('The ILQC algorithm found a solution in %fs \n\n',t cpu); 47 fprintf('Final Quadcopter Position: xQ = %.3f, yQ = %.3f, zQ = %.3f \n',sim\_out\_ilqc.x(1:3,end)); 48 fprintf('Final Quadcopter Velocity: xQ = %.3f, yQ = %.3f, zQ = %.3f \n'.sim out ilgc.x(7:9.end)); 49 -Visualize(sim out ilgc,Model.param,'plot mode',1); 50 % Plot Result(sim out ilgc,Model.param,'plots',plotvec(plot ind),'file',create pdf(plot ind),'path',pwd)

#### **ETH** zürich

#### main\_ex1.m:



📝 Edito	r - /home/winklera/git/MATLAB/l_olcar_2015_ex1_code/main_ex1.m
🗄 🗌 mair	n_ex1.m 🗙   ILQC_Design.m 🗙   LQR_Design.m 🗙   Cost_Design.m 🗙   Task_Design.m 🗙   🕂
1	% OLCAR - Exercise 1 - ILQC
2 -	close all; clearvars; clc;
3	
4 -	addpath(genpath(pwd)); % adds folders and subfolders to path
5	% To generate plots of LQR/ILQG rollout
6 -	<pre>plotvec = {'quad_pos_noLoad','quad_angles_noLoad','control_input_noLoad','rptor_thrust_noLoad';;</pre>
7 -	create_pdf = [0 0 0 0 ]; % for which plots should a pdf be created
8 -	plot_ind = [1 2 3 4 ]; % which data to plot on screen
9	
10	on the definition
11	Tack definition
12 -	Task = Task_Design();
1.0	Stand the dynamic model of the guadeanter
15 -	load the dynamic model of the quadcopter
16	coad quarters - moder , so save as structure moder
17	% Define cost function
18 -	Task.cost = Cost Design( Model.param.m0. Task ):
19	
20	
21	
22	% Problem 1: Initial controller design
23 -	[Initial_Controller, Cost_LQR] = LQR_Design(Model, Task);
24	
25	% Visualization of LQR controller
26 -	sim_out_lqr = Quad_Simulator(Model,Task,Initial_Controller);
27 -	disp('LQR controller performance:');
28 -	fprintf('Cost with LQR controller (metric: LQR cost function!): J* = %.3f \h', Cost_LQR);
29 -	<pre>fprintf('Start Quadcopter position: x = %.3f, y = %.3f, z = %.3f \n', sim_out_lqr.x(1:3,1));</pre>
30 -	<pre>fprintf('Final Quadcopter position: x = %.3f, y = %.3f, z = %.3f \n\n', sim_out_lqr.x(1:3,end));</pre>
31 -	Visualize(sim_out_lqr,Model.param,'plot_mode',1);
32	<pre>% Plot_Result(sim_out_ldr,Model.param,'plots',plotvec(plot_ind),'file',create_pdf(plot_ind),'path',pwd)</pre>
33	
34	see commont out to proceed to Droblem 2
36 -	refure.
37	
38	
39	% Problem 2: ILOC controller design
40 -	t cpu = cputime:
41 -	[ILQC Controller, Cost] = ILQC Design(Model,Task,Initial Controller,QQuad Simulator);
42 -	t_cpu = cputime - t_cpu;
43	
44	% Visualization of ILQC controller
45 -	<pre>sim_out_ilqc = Quad_Simulator(Model,Task,ILQC_Controller);</pre>
46 -	fprintf('The ILQC algorithm found a solution in %fs \n\n',t_cpu);
47 -	<pre>fprintf('Final Quadcopter Position: xQ = %.3f, yQ = %.3f, zQ = %.3f \n',sim_out_ilqc.x(1:3,end));</pre>
48 -	<pre>fprintf('Final Quadcopter Velocity: xQ = %.3f, yQ = %.3f, zQ = %.3f \n',sim_out_ilqc.x(7:9,end));</pre>
49 -	Visualize(sim_out_ilqc,Model.param,'plot_mode',1);
50	% Plot_Hesult(sim_out_ilqc,Model.param,'plots',plotvec(plot_ind),'file',create_pdf(plot_ind),'path',pwd)



## **Problem 1.1: Compute LQR feedback gain K**

- Linearize system dynamics (A,B):  $\dot{\mathbf{x}} = \mathbf{F}(\mathbf{x}, \mathbf{u}) = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}$   $\downarrow e.g. Model.Alin{1}(x_lin,u_lin, Model.param.syspar_vec),...$  $\dot{\mathbf{x}} = \mathbf{A}_{lin}(\mathbf{x}_{lin})\mathbf{x} + \mathbf{B}_{lin}(\mathbf{u}_{lin})\mathbf{u}$
- Cost function (Q,R)
  - (given in Cost\_Design.m)
- Matlab



### **Problem 1.2: Controller structure**

- For every time-step n: Feedback and feedforward elements combined in  $\theta_n \in \mathbb{R}^{13 \times 4}$ 



•  $\theta \in \mathbb{R}^{13 \times 4 \times N_t}$ 



## **Problem 1.2: Design controller from K**

- Task: Move quadrotor to goal position at  $\mathbf{x}_d(1) = 10m$ 
  - Define metric LQR controller tries to regulate to zero
  - $\mathbf{u} = K(\dots$
  - Incorporate that in controller structure (design  $\theta$ )
- Run and observe...
  - (Change cost matrices, goal positions, metric,...)



### **Problem 1.3: Include via-points**

- Task: Move quadrotor through specific states before reaching the goal state.
- Time-varying control law
  - $\theta_n$  changes over time
- Observe and run





# Problem 1 ✓

### Finished designing LQR controller

- Protected function "LQR\_Design\_Solution.p" available
- → Adapt main\_ex1.m.



#### **ETH** zürich

		Z Editor - /home/winklera/git/MATLAB/l_olcar_2015_ex1_code/main_ex1.m
		: main_ex1.m x ILQC_Design.m x LQR_Design.m x Cost_Design.m x Task_Design.m x +
		1 % OLCAR - Exercise 1 - ILOC
		2 - close all: clearvars: clc:
		3
		4 - addpath(genpath(pwd)): % adds folders and subfolders to path
		5 % To generate plots of LOR/ILOG rollout
		6 - plotvec = {'guad pos noLoad'.'guad angles noLoad'.'control input noLoad'.'rotor thrust noLoad'};
		7 - create pdf = [0 0 0 0]: % for which plots should a pdf be created
		8 - plot ind = [1 2 3 4 ]; % which data to plot on screen
		9
		10
		11 % Task definition
		12 - Task = Task Design();
		13
		14 %Load the dynamic model of the quadcopter
		15 - load('Quadrotor Model.mat','Model'); % save as structure "Model"
		16
		17 % Define cost function
		18 - Task.cost = Cost_Design( Model.param.mQ, Task );
		19
		20
		21
_	LOD Declara	22  S Problem 1: Initial controller design
	I UR Design V	23 - [Initial_Controller, Cost_LQR] = LQR_Design(Model, Task);
		24
		25 % Visualization of LQR controller
		<pre>26 - sim_out_lqr = Quad_Simulator(Model,Task,Initial_Controller);</pre>
		27 - disp('LQR controller performance:');
		<pre>28 - fprintf('Cost with LQR controller (metric: LQR cost function!): J* = %.3f \h', Cost_LQR);</pre>
		29 - fprintf('Start Quadcopter position: $x = \%.3f$ , $y = \%.3f$ , $z = \%.3f \n'$ , sim_out_lqr.x(1:3,1));
		<pre>30 - tprintt('Final Quadcopter position: x = %.3t, y = %.3t, z = %.3t \n\n', sim_out_lqr.x(1:3,end));</pre>
		<pre>31 - Visualize(sim_out_lqr,Model.param,'plot_mode',1);</pre>
		32 % Plot_Result(sim_out_lqr,Model.param,'plots',plotvec(plot_ind),'file',create_pdf(plot_ind),'path',pwd)
		33
		34
		35 %% comment out to proceed to Problem 2
		36 - return;
		3/
	$\Delta \parallel O \cap D$	38
		39 %% Problem 2: ILQC controller design
	· 0	40 - [JUG controllor, Cost] - ILCC Design(Medel Task Initial Controllor @Oued Simulator);
		41 - trouter, costi - trouter, costi - trouter, task, intrat_controtter, equal_similatory,
		42 - L_CPU - CPULLING - L_CPU,
		43 AA & Visualization of LLOC controller
		45 - sim out ilco - Quad Simulator (Model Task ILOC Controller).
		46 - forintf('The I loc algorithm found a solution in %fs \n\n', t cou):
		47 - forintf('Final Quadcopter Position: $x0 = %.3f$ , $y0 = %.3f$ , $z0 = %.3f$ \n'.sim out iloc.x(1:3 end)).
		48 - forintf('Final Quadcopter Velocity: x0 = %.3f, y0 = %.3f, z0 = %.3f \n', sim out ilcc.x(7:9.end)):
		49 - Visualize(sim out ilgc.Model.param.'plot mode'.1):
		50 % Plot Result(sim out ilgc.Model.param.'plots'.plotyec(plot ind).'file'.create pdf(plot ind).'path'.pwd)

### A D R L OLCAR – Exercise 1

## **Problem 2.1: Design ILQC Controller**

- LQR: system dynamics linearized around one linearization point
  - $\rightarrow$  linear time-invariant system
- ILQC: system dynamics linearized around each discretized state (50Hz)
  - $\rightarrow$  linear time-variant system

LQC	Design.m
LQU_	

ΪE	ditor	r - /home/v	vinklera/git/MA	TLAB/l_olcar_2	015_ex1_	code/Design_fu	uncti	ons/ILQC_Desig	jn.m	
ſ	main_	_exl.m 🗶	ILQC_Design.m	_x LQR_Desi	gn.m 🗙	Cost_Design.m	X	Task_Design.m	×ŀ	+
88										
89		%% F	Problem 2.1.2:	Solve Ricca	ti-like	equations ba	ickwa	rds in time		
90		% Ir	itialize the	value functi	on elem	ents starting	, at	final time s	tep	
91		% (E	Eq.(1.87)							
92	-	xf. =	= <u>XQ</u> (:,end); %	s final state	e when u	sing current	cont	roller		
93		% Sn	n(:,:,n_t) = .							
94		% S\	$i(:,n_t) = .$							
95		% S(	(n_t) = .							
96										
97		% "E	Backward pass"	: Calculate	the coe	fficients (s,	Sv,S	Sm) for the v	alue	
98		% TL	inctions at ea	arlier times	by proc	eeding backwa	irds	in time (DP-	approa	ach)
99	- 5	a for	n = (length(s	sim_out.t)-1)	:-1:1					
00										
01			% state of sy	stem at time	step n					
02	-		$x \omega = x \omega(:, n);$							
03	-		$u \alpha = u \alpha(:, n);$							
04										
05			% Discretize	and lineariz	e conti	nuous system	ayna	mics Alin ar	bund	
00			* specific pa	air (x0,00).	See exe	rcise sneet E	-q.(4	<pre>#) for details</pre>	5	
07			≈ A =;							
08			ж B =;							
109										
10			% and onder a	opposi mati op	of cos	t function at	+ + i =	o otop p (Ea	(1 70	
12				approximation		t function at		ie step n (Eq	. (1.76	, ,
12			* Qy							
14			% Qv =							
15			% Qui							
16			% Rm =							
17			% Dm =							
18			· · · · · · · · · · · · · · · · · · ·							
19			% control der	endent terms	of cos	t function (F	a. (1	.81))		
20			% a =		% l	inear control	der	endent		
21			% G =		% C	ontrol and st	ate	dependent		
22			% H =		% a	uadratic cont	rol	dependent		
23								1.1		
24			% ensuring H	remains symm	netric					
25	-		H = (H+H')/2;	% important	, do no	t delete!				
26										
27			% the optimal	. change of t	he inpu	t trajectory	du =	= duff + K*dx	(Eq. (	(1.82)
28			% duff(:,:,n)	=						
29			% K(:,:,n)	=						
30										
31			% Solve Ricca	ati-like equa	itions f	or current ti	me s	step n (Eq.(1	.84)	
32			% Sm(:,:,n) =	=						
33			% Sv(:,n) = .							
34			% s(n) =							
35										
36	-	- end	% of backward	l pass for so	lving R	iccati equati	on			
37										
38		% de	stine theta_ff	in this fur	iction					
39	-	Cont	roller.theta	= Update_Cor	troller	(X0,U0,duff,K	0;			
40										
41	-	1 =	1+1;							



## **Problem 2.1: Design ILQC Controller**

#### (Discretization) :

$$\begin{split} \dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} & \text{(continous system dynamics)} \\ \Leftrightarrow \frac{\mathbf{x}_{n+1} - \mathbf{x}_n}{\delta t} = \mathbf{A}\mathbf{x}_n + \mathbf{B}\mathbf{u}_n & \downarrow \\ \Leftrightarrow & \mathbf{x}_{n+1} &= (\mathbf{I} + \mathbf{A}\delta t)\mathbf{x}_n + (\mathbf{B}\delta t)\mathbf{u}_n \\ \Leftrightarrow & \mathbf{x}_{n+1} &= \mathbf{A}_n\mathbf{x}_n + \mathbf{B}_n\mathbf{u}_n & \text{(discrete system dynamics)} \end{split}$$

- Run and observe...
- Protected function ILQC\_Design\_Solution.p available



#### **ETH** zürich

#### main\_ex1.m:



📝 Edi	🜠 Editor - /home/winklera/git/MATLAB/l_olcar_2015_ex1_code/main_ex1.m				
i ∫ ma	ain_ex1.m 🗙 ILQC_Design.m 🗙 LQR_Design.m 🗙 Cost_Design.m 🗙 Task_Design.m 🗶 🕂				
1	% OLCAR - Exercise 1 - ILQC				
2 -	close all; clearvars; clc;				
3					
4 -	addpath(genpath(pwd)); % adds folders and subfolders to path				
5	% To generate plots of LQR/ILQG rollout				
6 -	<pre>plotvec = {'quad_pos_noLoad', 'quad_angles_noLoad', 'control_input_noLoad', 'rotor_thrust_noLoad'};</pre>				
7 -	create pdf = [0 0 0 0]; % for which plots should a pdf be created				
8-	plot_ind = [1 2 3 4 ]; % which data to plot on screen				
9					
10	& Tack definition				
12 -	Tack = Tack Design().				
13	Task = Task_beargn(),				
14	%Load the dynamic model of the quadcopter				
15 -	load('Quadrotor Model.mat','Model'); % save as structure "Model"				
16					
17	% Define cost function				
18 -	Task.cost = <u>Cost_Design</u> ( Model.param.mQ, Task );				
19					
20					
21					
22	SA Problem 1: Initial controller design				
23 -	[Initial_Controller, Cost_LQR] = LQR_Design(Model, Task);				
24	a visualization of LOD controller				
25	% Visualization of EQR controller				
20 -	dip(ULPR_controller_performance))				
28 -	for int('Cost with LOB controller (matric: LOB cost function!): 1* = % 3f \b' Cost LOB).				
29 -	$f_{\text{printf}}(Start Quadconter position: x = 8.3f, y = 8.3f, z = 8.3f, y, sim out loc.x(1:3.1)):$				
30 -	for int ('Final Quadcopter position: $x = $ %.3f, $y = $ %.3f, $z = $ %.3f (\\\\', sim out lar.x(1:3.end)):				
31 -	Visualize(sim out lgr,Model.param,'plot mode',1);				
32	<pre>% Plot_Result(sim_out_lqr,Model.param,'plots',plotvec(plot_ind),'file',create_pdf(plot_ind),'path',pwd)</pre>				
33					
34					
35	🏶 comment out to proceed to Problem 2				
36 -	return;				
37					
38					
39	% Problem 2: ILQC controller design				
40 -	t_cpu = cputime; [I.O.C. Controllor, Cost] = I.O.C. Design(Model Task Initial Controllor @Quad Simulator);				
41 -	t cou controller, cost = itoc_besign(Model, ask, initiat_controller, @dad_simulator),				
43	c_opu = opucine - c_opu,				
44	% Visualization of TLOC controller				
45 -	<pre>sim out ilgc = Quad Simulator(Model,Task,ILQC Controller):</pre>				
46 -	<pre>fprintf('The ILQC algorithm found a solution in %fs \n\n',t cpu);</pre>				
47 -	<pre>fprintf('Final Quadcopter Position: xQ = %.3f, yQ = %.3f, zQ = %.3f \n',sim_out_ilqc.x(1:3,end));</pre>				
48 -	<pre>fprintf('Final Quadcopter Velocity: xQ = %.3f, yQ = %.3f, zQ = %.3f \n',sim_out_ilqc.x(7:9,end));</pre>				
49 -	<pre>Visualize(sim_out_ilqc,Model.param,'plot_mode',1);</pre>				
50	<pre>% Plot_Result(sim_out_ilqc,Model.param,'plots',plotvec(plot_ind),'file',create_pdf(plot_ind),'path',pwd)</pre>				



## **Problem 2.2: Include via-points**

- Append ILQC cost function
  - Penalizes deviation from via point  $\mathbf{x}_{vp}$  only in proximity of time  $t_{vp}$ .

$$L_{vp}(t) = (\mathbf{x} - \mathbf{x}_{vp})^{\top} \mathbf{Q}_{vp} \cdot \sqrt{\frac{\rho}{2\pi}} e^{\left(-\frac{\rho}{2}(t - t_{vp})^2\right)} \left(\mathbf{x} - \mathbf{x}_{vp}\right)$$

- Run and observe
  - Use different via points
  - Compare to LQR

[Cedric de Crousaz, Farbod Farshidian, and Jonas Buchli. Aggressive optimal control for agile flight with a slung load. Workshop, IROS 2014]



#### ETHzürich



