REINFORCEMENT LEARNING FOR ADAPTIVE CONTROL OF QUADRUPEDAL ROBOTS: AN EMPERICAL STUDY

Yichao Zhong
ACM Class
Shanghai Jiao Tong University

ABSTRACT

The adaptivity has been a critical characteristic that is utilized by humans and animals to complete more tasks in better, more cozy, more comfortable approaches that will lead to lower damage. In the topic of Robotics and Reinforcement Learning, adaptivity is also of necessity and is worth researching how to attain and maintain the adaptivity of policies. In this paper, we declare and validate the findings that a previous work, Rapid Motion Adaptation (RMA), is totally robustness and has little adaptivity. We find that the RMA policies work well even if their adaptation module output is pure noise. We propose to use auxiliary losses and a pretrained 'world model' to relieve this severely bad phenomenon. Further, we put forward that applying complex network structures helps the learning of adaptivity, just letting the teacher-student framework out. We also investigate if better history utilization would work and give several insights on how to maintain adaptivity. More, we deploy our methods on real quadrupedal robot Unitree A1 and it valids in real world too.

1 INTRODUCTION

In the realm of robotics, the pursuit of adaptivity stands as a cornerstone in the quest for machines that can seamlessly navigate the complexities of various environments and tasks. Just as animals and humans leverage their adaptivity to conquer diverse challenges, the need for robots to possess and maintain this essential trait becomes increasingly apparent. It is within this context that this paper delves into the crucial realm of maintaining adaptivity for quadruped robots through the lens of Reinforcement Learning, presenting a pivotal foundation for the advancement of robotic capabilities.

Well, adaptivity and robustness are always a trade-off in the concept of learning. As for Robot Learning, robustness have been enhanced to maximum through domain randomizations, including frictions, additional base mass, controller stiffness, how hard to push the robot, etc. Since the way to robustness is known, the way to adaptivity, however, seems ever more critical.

In the ensuing sections of this emperical study, we review the state-of-the-art work in this content, RMA, discuss adaptation within a validation study, and invalidate the adaptivity of RMA policy: we find that adding severe noise to the adaptation module in RMA does not effect the performance, indicating that RMA is nothing but robustness. We then put forward some better solutions for gaining and maintaining adaptivity to achieve better performances. And we do sufficient both sim and real experiments to validate or invalidate the novel thoughts. Literally, our major contributions in this work are:

- We invalidated RMA's adaptivity.
- We built auxiliary losses to overcome the mentioned over-robust phenomenon in RMA and also reached higher performances.
- We combined complex architectures and different history utilizations with reinforcement learning to attain better adaptivity.

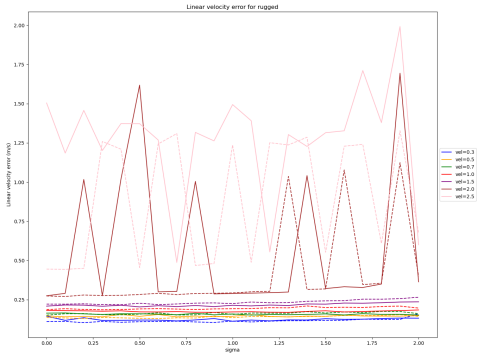


Figure 1: RMA noise test. Dashed lines for RMA-Teacher and Solid lines for RMA-student. We tested the linear velocity tracking error under various velocity commands and uniform noise scales levelled 0 to 2. The noises hardly change the performances which shows the excellent robustness of RMA base policy but nothing adaptive.

As we stand at the precipice of a new frontier in robotics, the pursuit of adaptivity for quadruped robots through the prism of Reinforcement Learning stands as an indomitable testament to our collective aspiration for machines that can seamlessly evolve and navigate an ever-changing world. In this spirit, this paper seeks to ignite a transformative discourse that champions the indelible significance of adaptivity, heralding a future where the adaptive prowess of robots rivals the very essence of nature itself.

2 Related Works

2.1 Rapid Motion Adaptation

Rapid Motion Adaptation **?** proposed a method of attaining adaptivity through teacher-student pipeline. The teacher policy is composed of a current privileged observation encoder and a base policy, and they are trained together. The student policy uses the same fixed base policy as the teacher's, and learns a non-privileged observation history encoder which should be align with the teacher encoder's output. It is learned by a MSE Loss between teacher and student's outputs.

3 Methodologies

3.1 Effect of training pipeline: reinvestigate RMA adaptivity

In this section we reinvestigate RMA's adaptivity. An adaptive policy, say, in RMA structure, definitely relies on its adaptation module. It's quite suprising to find out that RMA's both teacher and student policy still functions well even if we add severe noises to the adaptation module, as our experiment result Fig.1 shows. Note that we regularized the latent vector to $[-1, 1]$ through a Tanh layer at the bottom of the net. The noise added is uniform noise, leveraging

$$z \longleftarrow clip(-1, 1, z + U(-\sigma, \sigma)) \ ,$$

where σ denotes the noise level. From the formula, it can be inferred that when $\sigma = 2$ it is totally pure noise. However, RMA-teacher and student has little evidence that they're effected by the noises on adaptation module; contrarily, the result shows that the base policy of RMA is robust enough. Let's describe such policies as "over-robust". Also, it shows the low transfer ability of RMA-student as the performance becomes totally bad in unseen conditions. If it's really an adaptive policy, the domain transferring won't be that kind of struggling because the policy and its systematic identification would change over different environments.

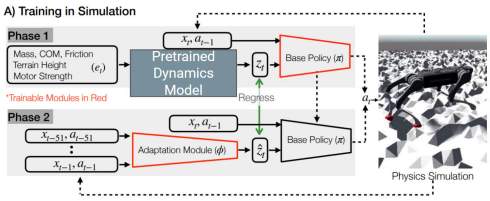


Figure 2: Auxiliary loss aided version of RMA. A fixed pretrained dynamics model replaces the environmental factor encoder.

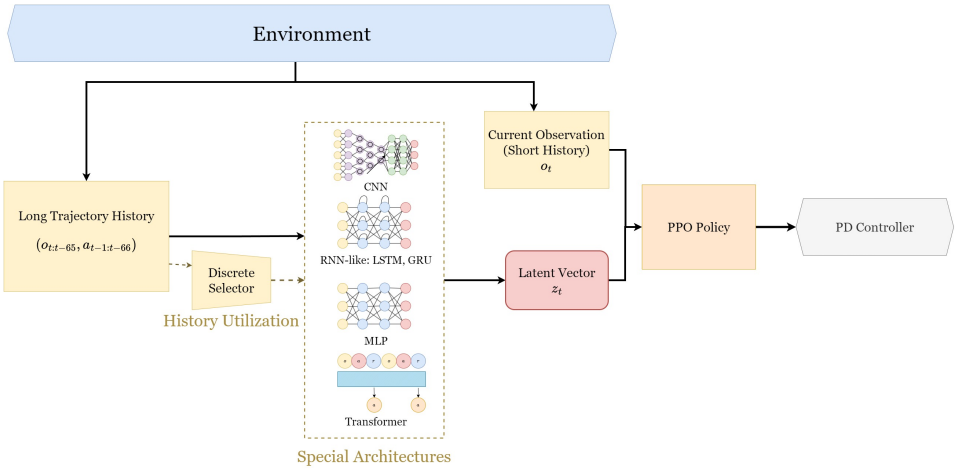


Figure 3: Dual-history structure with complex long history encoder adopted. A dashed line means that it could be ignored.

3.2 Auxiliary Losses

To avoid the "over-robust" phenomenon, we are suspicious that the deviation happened in the first training phase in RMA when we used RL loss to train both the base policy and the environmental factor encoder. We cannot promise that both two networks are trained as expected and z_t could be really representing as extrinsic factors well, because z_t hasn't got any constraints to ensure its representing ability. z_t could be anything.

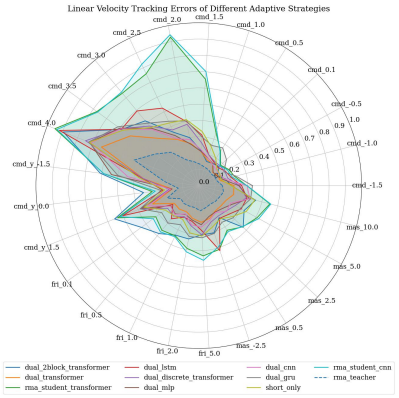
Still maintaining the teacher-student pipeline, we build some auxiliary losses for aid. As We change the environmental factor encoder to be a pretrained dynamics model ϕ, which is trained under the reconstruction loss, like auto-encoder, or next-step prediction loss. Here, for instance, we use the next-step prediction loss. The pretrained dynamics model ϕ is totally frozen during training teacher and student. To learn the transition process of dynamics, we first use a random policy to explore in Isaacgym and collect the data $D = \{o_t, a_t\}$. The loss is depicted as:

$$L_{pred} = MSE(f \circ o_t, \psi(\phi(o_t, a_{t-1}))) \ ,$$

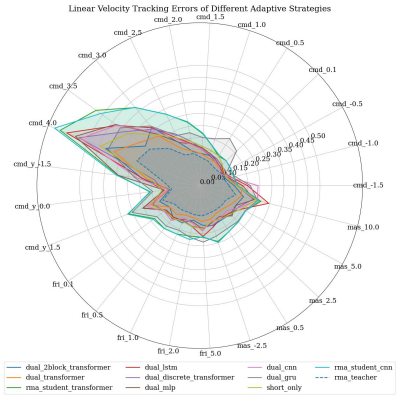
where f is a $[0, 1]^{obs}$ vector that selects the important observations like dof position and dof velocity as the objective to predict and ψ serves as the decoder in auto-encoder structure. This encoder at least learns some transition features of the environment, and its representing ability can be assured which should be better than the RMA encoder.

3.3 Network Structures Application

Apart from teacher-student pipeline, another way to attain adaptivity is the application of complex architectures which is complex enough to learn the adaptivity from history trajectories. Accounting for the threshold of inference time and the trainability in RL, it shouldn't be too complex. Take Li et al. (2024)'s idea, we adopt the dual-history structure. Li et al. (2024) just assessed the CNN network and we are going to do more, referring to 3.



(a) Evaluation on rough, smooth, stair and discrete terrains



(b) Evaluation on rough terrains only

Figure 4: Results for all the methodologies we tested on our metric for adaptivity. The linear tracking velocity errors (m/s) in simulation for different adaptive methods, including improvement on architecture, history utilization, teacher-student, and auxiliary loss design. The radius is the linear tracking velocity error we measure in sim in m/s. And each angle represents a special environment condition or instruction. 'Cmd' refers to given command x(or y)-axis velocity(m/s). 'Fri' refers to friction. 'Mas' refers to the base mass added. The smaller in area means the policy has better adaptivity.

As 3 describes, this training is free from teacher-student pipeline. The input contains two parts of history, a short one (regarded as current observation) and a long one, where the complex structures would learn the system identification from. We tried different kinds of networks: CNN, GRU, LSTM, MLP and decision transformerChen et al. (2021)(one block and two block). Note that we use the output of transformer (predicted action in decision transformer) directly as the latent vector. We just regard the transformer as a complex network.

3.4 Do we need special history designs

The history could be specially designed, like you don't have to record the whole history, from $t-1$ to $t-h$, it's like that you might take some discrete pieces of history, and there lies a lot of things to be investigated. The discretized history, for instance, to be 2-powered arrays like $o_{t-1}, o_{t-2}, o_{t-4}, o_{t-8}, o_{t-16}, ...$, might take less spaces and speed up computation simultaneously, which is quite beneficial for real-time locomotions. But we didn't make it to real-world deployment for this project.

4 Experimental Results

4.1 In simulation

Fig. 4 presents an all-rounded comparison on choosing different network structures as our long-history model. To assess the performance of the policies throughout various environments, we rolled out 25 metrics under different circumstances for each policy: 15 for command velocities, 5 for added mass, and 5 for frictions. We drew radar plots to better visualize the policies' overall ability. For each point, the distance to the origin point represents the linear velocity tracking error, serving as our main metric in this study. And each angle represents a specific environmental condition. Detailed metric data refers to 4. Table 1 shows the default conditions and metric settings, and each metric only alters what it denotes. More, for fair comparisons, all the methodologies are trained under the same hyperparameters.

Parameter	Value
history length	66
latent dimension	12
x-axis command velocity range	[-1.0,3.0] m/s
y-axis command velocity range	[-1.0,1.0] m/s
friction range	[0.5,1.25]
base mass range	[0,2.5] kg
stiffness	28
damping	0.7
training iteration	20000

Table 1: The default settings in evaluation

Label examples	Explanations
cmd-0.5	x-axis velocity command=0.5m/s
cmd-y-0.5	y-axis velocity command=0.5m/s
mas-0.5	added base mass=0.5kg
fri-0.5	friction coefficient in the environment=0.5

Table 2: The specific settings in evaluation

Network Architecture We compared the choices of the long-history network architecture. This network analyzes systematic identification from a relatively long trajectory history, so we tried out using convolutional models like CNN, sequential models like LSTM, GRU and transformers, and basic models like MLP. Note that all of them takes in a fixed length of history trajectory as their inputs. And all of the networks are trained directly by RL with the same fixed training iterations.

As in Fig. 4, the policy using a one-block transformer as a long-history encoder has a lower tracking error than all others in almost every situation. It shows great adaptivity to high-speed and low-speed situations, different levels of pressure, and slippy and rough terrain. Note that this one-block transformer takes in trajectories as tokens in the format of $\{s_{t-h+1}, a_{t-h}, \ldots, s_{t-1}, a_{t-2}, s_t, a_{t-1}\}$ where h denotes the history length, which is identical to Decision Transformer.

Note that the one-block transformer outperforms the two-block transformer. Despite of the simplicity, the one-block transformer achieves competitive results as it has the least area among all the methodologies except for the privileged teacher. So we shall compare other methodologies with the one-block transformer later.

History Utilization We compare the one-block full-length transformer with one-block discrete transformer. The discrete transformer takes several discrete history steps as input, rather than the whole. For instance, we design the discrete history as the concatenation of $\{o_{t-64}, o_{t-32}, o_{t-16}, o_{t-8}, o_{t-4}, o_{t-1}, o_t\}$ where $o_t = \{s_t, a_t\}$.

Table 4 and 5 shows that a simplified design for history leads to a general loss in performance. However it is still quite competitive; a reduction in history length speeds up the computation and takes less inference time, which is critical for Sim2Real.

To validate the use of history is of necessity, we also tried without the long history, literally "short only" in 4 and 5. It gets truly less competitive without the input of long history.

Training Pipeline Design Do we need a special training pipeline design like teacher-student for adaptivity? That's been a continuous dispute. And in this section, we choose state-of-the-art RMA teacher-student structure for comparison. Note that we use the same history length for both the transformer-based policy and the rma-student. Table 4 and 5 show the adaptivity of our one-block transformer policy over the RMA-student. The performance of the one-block transformer policy does not surpass RMA-teacher, which is regarded as an upper bound of all non-privileged policies, further validating our results.

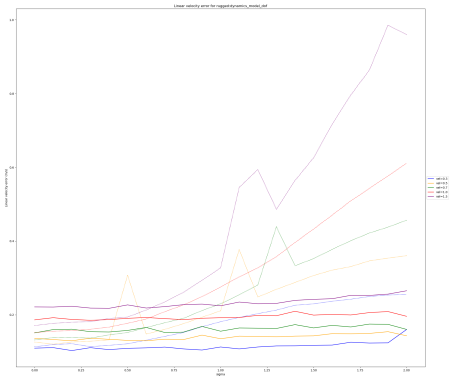


Figure 5: Dashed lines for our auxiliary prediction loss-aided teacher and Solid lines for RMA teacher. Focus on the 0-noise condition, ours performs better than RMA Teacher. and it could be easily effected by noises on the encoder output, therefore further validating the adaptivity it learns.

	RMA student Policy	Transformer Dual Policy
Mean speed error(%)↓	26.98 ± 10.16	10.38 ± 5.637
Success Rate (%)	100	100

Table 3: Real world deployment metrics. Quadrupedal Robot Unitree A1 walks a 2m distance on the plane with a consistent forwarding command of 0.5m/s. We measure the time consumed and calculate the mean velocity over 3 laps, forwards and backwards.

Auxiliary Loss Design for Adaptivity As mentioned, we found that RMA-student still works well even when its adaptation module output is replaced by random numbers. And 1 evidently sheds light on this discovery. So the adaptation module in RMA is of no use most of the time. We observe that RMA-teacher also suffers from this problem, indicating that this teacher-student pipeline trains a much too robust base policy and invalid precedent encoders. The source of the issue lies in the teacher policy training session.

So, to overcome the issue, we built some auxiliary loss designs for the environment factor encoder. We tried applying the prediction loss, which is the error against the next step factors. The factors to predict are the DoFs' positions and velocities. Fig. 5 shows that they both reach better performance than teacher-student under zero-noise conditions. And as Fig. 5 shows, it passes the adaptation module validation test.

4.2 Real-world Experiments for A1

Velocity tracking performance We have also deployed our most competitive method, which applies dual history and transformer structure, onto the Unitree A1 quadrupedal robot. Our main concern is the velocity tracking error. We evaluated them by walking on the plane with a command 0.5m/s for a distance of 2 meters and we tested the time consumed. And the result is as 3 shows.

5 Conclusion

We have drawn to our conclusion that the vanilla dual history structure can attain higher adaptivity than RMA student, and on Unitree A1 the one-block transformer works the best. More, history utilization skills like discretize histories does not improve the performances, so does not using history. And the aid of auxiliary prediction loss for the teacher encoder do prevent the teacher to learn to be too robust in teacher-student pipeline. However the teacher-student pipeline is no longer critical for adaptivity because it does not train a student policy that is better than the transformer-structured policy.

ACKNOWLEDGMENTS

First, I'd like to convey a big thanks to Tairan He, Carnegie Mellon University Ph.D, who helped me throughout this project a lot I also fully appreciate my advisor, Prof. Weinan Zhang, who provided the two quadruped robots Unitree A1, for my real-world experiments. Moreover, I'm grateful for the countless help and aid from my labmates in the APEX Data and Knowledge Management Lab Robotics Group: Hang Lai, Jiahang Cao and Wentao Dong. Also, thanks to Prof. Yong Yu who has always been guiding me throughout my University life.

REFERENCES

Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Michael Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling, 2021.

Ashish Kumar, Zipeng Fu, Deepak Pathak, and Jitendra Malik. Rma: Rapid motor adaptation for legged robots, 2021.

Zhongyu Li, Xue Bin Peng, Pieter Abbeel, Sergey Levine, Glen Berseth, and Koushil Sreenath. Reinforcement learning for versatile, dynamic, and robust bipedal locomotion control, 2024.

A APPENDIX

A.1 DETAILED EXPERIMENT RESULTS

See below.

Table 4: Linear Velocity Tracking Errors of Different Strategies in All Terrains. The numbers are the average tracking errors of the trajectories. The smaller the better.

	Dual-TF	Dual-2BTF	Dual-MLP	Dual-CNN	Dual-LSTM	Dual-GRU	**RMA Teacher (Expert)**	RMA-Student CNN	RMA-Student TF	Dual-DiscreteTF	No Hist
cmd v_x											
-1.5	0.2162	**0.1992**	0.2456	0.2959	0.2633	0.3296	0.1452	0.2964	0.2948	0.2517	0.1989
-1.0	**0.1352**	0.1674	0.1536	0.1632	0.2051	0.2357	0.1160	0.2117	0.2276	0.1837	0.1436
-0.5	0.1324	0.1313	**0.1252**	0.1449	0.1854	0.1755	0.1123	0.2004	0.2079	0.1827	0.1467
0.1	**0.1181**	0.1713	0.1208	0.1449	0.1623	0.2332	0.1048	0.1781	0.1832	0.1779	0.1518
0.5	**0.1587**	0.1610	0.1607	0.1562	0.1718	0.2744	0.1138	0.2444	0.2500	0.1777	0.1960
1.0	0.1661	**0.1557**	0.1796	0.1818	0.1990	0.2593	0.1191	0.3568	0.3422	0.1992	0.2498
1.5	**0.2102**	0.2156	0.2111	0.2212	0.3107	0.3001	0.1316	0.7014	0.6546	0.2642	0.3331
2.0	**0.2633**	0.3002	0.2585	0.2875	0.3999	0.4045	0.1489	0.9432	0.9284	0.3834	0.4129
2.5	**0.3142**	0.3619	0.3393	0.3187	0.5258	0.4370	0.1745	0.8231	0.7590	0.3781	0.4250
3.0	**0.3760**	0.5160	0.4578	0.4466	0.5951	0.4971	0.2659	0.7448	0.7365	0.4745	0.4634
3.5	**0.5192**	0.5868	0.6000	0.5931	0.5880	0.6144	0.3301	0.7675	0.7736	0.5952	0.5780
4.0	**0.6429**	0.8947	0.7207	0.7126	0.9205	0.7273	0.4270	0.9372	0.9499	0.7481	0.7217
cmd v_y											
-1.5	**0.3428**	0.6102	0.5400	0.4399	0.3470	0.5010	0.2146	0.5897	0.5889	0.3429	0.3657
0.0	**0.1651**	0.3421	0.1655	0.1741	0.2228	0.2343	0.1275	0.2692	0.2881	0.1833	0.2115
1.5	**0.3047**	0.5558	0.3780	0.3372	0.5022	0.3881	0.2080	0.5203	0.5366	0.3856	0.3613
friction											
0.1	**0.1859**	0.4359	0.2122	0.2151	0.1968	0.2818	0.1367	0.3453	0.3260	0.2368	0.2548
0.5	**0.1871**	0.3823	0.2266	0.2304	0.2644	0.2980	0.1426	0.3695	0.3560	0.2430	0.2394
1.0	**0.1773**	0.3359	0.1950	0.1915	0.2140	0.2651	0.1384	0.3467	0.3402	0.2102	0.2298
2.0	**0.2043**	0.3333	0.2150	0.2452	0.2234	0.3178	0.1369	0.4132	0.3917	0.2519	0.2955
5.0	**0.2257**	0.3029	0.2408	0.2555	0.2764	0.3207	0.1550	0.4596	0.4322	0.3030	0.3106
base mass											
-2.5	**0.2123**	0.3049	0.1999	0.2227	0.4197	0.3087	0.1417	0.3944	0.4117	0.2371	0.2685
0.5	**0.1860**	0.2329	0.1910	0.1947	0.2381	0.2663	0.1341	0.3346	0.3246	0.2218	0.2429
2.5	**0.1842**	0.3740	0.2032	0.2323	0.2559	0.2891	0.1410	0.3678	0.3700	0.2286	0.2787
5.0	**0.1901**	0.3174	0.2362	0.2229	0.2968	0.2910	0.1454	0.4206	0.4089	0.2548	0.3316
10.0	**0.2207**	0.3604	0.3210	0.3253	0.3054	0.3286	0.1570	0.4565	0.4509	0.3017	0.3558

Table 5: Linear velocity tracking errors of different strategies in rough terrains only. Considering pacing upstairs and downstairs with the speed of more than 3.0m/s almost impossible, we derived their performances on rough terrains for references.

	Dual-TF	Dual-2BTF	Dual-MLP	Dual-CNN	Dual-LSTM	Dual-GRU	RMA Teacher	RMA-Student CNN	RMA-Student TF	Dual-Discrete TF	NoHist
cmd v_x											
-1.5	0.1768	0.1438	0.1789	0.2426	0.1986	0.2102	0.1217	0.1588	0.1580	0.1545	0.1520
-1.0	0.1183	0.1117	0.1266	0.1339	0.1367	0.1524	0.1052	0.1480	0.1462	0.1253	0.1174
-0.5	0.1105	0.1077	0.1073	0.0993	0.1129	0.1320	0.1041	0.1459	0.1466	0.1232	0.1155
0.1	0.1095	0.1167	0.1157	0.1127	0.1232	0.2102	0.0950	0.1513	0.1512	0.1243	0.1287
0.5	0.1174	0.1197	0.1253	0.1206	0.1319	0.2317	0.0977	0.1508	0.1458	0.1322	0.1462
1.0	0.1195	0.1317	0.1394	0.1364	0.1335	0.2049	0.1082	0.1727	0.1714	0.1406	0.1537
1.5	0.1324	0.1334	0.1586	0.1511	0.1505	0.2003	0.1164	0.2181	0.2155	0.1535	0.1641
2.0	0.1470	0.1437	0.1771	0.1737	0.1775	0.2240	0.1362	0.2704	0.2677	0.1873	0.1867
2.5	0.1687	0.1959	0.1981	0.2120	0.2184	0.2302	0.1407	0.3275	0.3286	0.2378	0.2194
3.0	0.2678	0.3143	0.2776	0.2894	0.3128	0.3219	0.2012	0.4222	0.4213	0.3051	0.2743
3.5	0.3074	0.2797	0.4331	0.4273	0.4256	0.4089	0.2580	0.5104	0.5330	0.3825	0.3703
4.0	0.3846	0.4209	0.5569	0.5334	0.5957	0.4107	0.2816	0.6499	0.6249	0.5414	0.4445
cmd v_y											
-1.5	0.2217	0.2430	0.3414	0.2651	0.2397	0.2737	0.1559	0.3504	0.3422	0.1897	0.2028
0.0	0.1300	0.1262	0.1488	0.1335	0.1505	0.1972	0.1182	0.2061	0.2096	0.1570	0.1589
1.5	0.2198	0.1793	0.2560	0.2361	0.2540	0.3033	0.1652	0.3233	0.3191	0.2097	0.2212
friction											
0.1	0.1529	0.1537	0.1743	0.1824	0.1724	0.2204	0.1303	0.2392	0.2331	0.1806	0.1781
0.5	0.1540	0.1743	0.1796	0.1876	0.1709	0.2188	0.1274	0.2322	0.2313	0.1774	0.1758
1.0	0.1360	0.1717	0.1552	0.1512	0.1610	0.2055	0.1207	0.2233	0.2230	0.1632	0.1651
2.0	0.1361	0.1459	0.1542	0.1597	0.1737	0.2193	0.1231	0.2283	0.2139	0.1728	0.1751
5.0	0.1501	0.1650	0.1812	0.1886	0.2120	0.2357	0.1267	0.2134	0.2161	0.1760	0.1797
base mass											
-2.5	0.1469	0.1752	0.1627	0.1550	0.1811	0.2214	0.1216	0.2483	0.2443	0.1734	0.1579
0.5	0.1341	0.1693	0.1520	0.1627	0.1619	0.2122	0.1207	0.2265	0.2265	0.1643	0.1647
2.5	0.1375	0.1740	0.1839	0.1528	0.1676	0.2174	0.1228	0.2208	0.2196	0.1790	0.1736
5.0	0.1550	0.1707	0.1714	0.1891	0.2015	0.2165	0.1244	0.2183	0.2171	0.1841	0.1983
10.0	0.1779	0.1881	0.2340	0.2475	0.2964	0.2480	0.1538	0.2566	0.2628	0.2297	0.2445