Improving Engine Performance Through MIMO Engine Airpath Control

Austin LaFever, Patrick Marlatt, Fred Peterson, Jonathan Wozny
FAMU-FSU College of Engineering

ABSTRACT

This paper describes the initial stages of using a multi-input multi-output (MIMO) control approach to control air flow components to improve transient torque tracking in a simulated turbocharged spark ignition engine in MathWorks' Powertrain Blockset. The specific components to be controlled are the throttle and the wastegate valves. Using a MIMO approach allows the parts that work together to be controlled together. This approach will use a set of linear model predictive controllers for the throttle and a set for the wastegate. These will be coordinated depending on the operating conditions (i.e. engine speed and torque command). Second order state-space models relating throttle to manifold absolute pressure (MAP) and wastegate to manifold absolute pressure were created using MathWorks' System Identification Toolbox; the torque command will be converted to a MAP command which will be referenced by the MPC controller. The actual implementation of the controller was not completed, but the state-space models and their performance in a model predictive controller under steady-state operating conditions was validated. Although the original goals were not completed, the process for creating internal models for the model predictive controllers and a framework for continuing the MIMO controller was successfully created.

INTRODUCTION

Internal combustion engines have found a wide range of applications in modern society. These engines rely on airflow through the engine from the outside environment in order to complete the combustion process and power the engine. Until now, internal combustion engines have relied on a single-input single-output (SISO) PI control approach developed decades ago. This approach manipulates the main components that control airflow into the engine using scheduled steady-state positions. During transient response, the output is not coordinated or optimal compared to steady-state operation, resulting in inefficient control of torque and emissions. Although effective steady state control is achievable, effective transient control with this method is not and therefore needs to be improved in order to keep up with increasingly competitive standards of engine performance and emissions.

In 2018, General Motors (GM) published an article discussing their success in using a MIMO control approach for transient and steady state operation of the air flow components of an internal combustion engine to achieve desired torque using model predictive control (MPC) [1]. Model predictive control is a control method that uses an internal model to predict future outputs to make the best control move. It handles MIMO systems well. Model predictive control requires a large amount of computing power and is only becoming a viable option to be run in engine control units (ECUs) recently as modern computing power has increased rapidly and the electric control components have become affordable for mass-implementation. GM was able to implement their controls in a production line electronic control unit (ECU) thanks to efficient production code that was able to run quickly on a production engine controller.

MathWorks' SI Engine Dynamometer in the Powertrain Blockset uses the standard SISO PI approach mentioned above. Torque and emissions do not respond well during the transient response, and the throttle and wastegate are not coordinated to provide a boost or intake manifold pressure (MAP) trajectory that results in good emissions and torque transient response. Our project goal is to follow the work of GM by implementing a model predictive control system for the air flow components, specifically the throttle and wastegate, to control transient and steady state torque and emissions effectively. The cur-

rent standard set of controls only make use of MIMO to produce coordinated and optimal control for output values during steady-state operating conditions, where our approach uses MIMO control of the air system to produce coordinated actuation in both transient and steady-state conditions. This control approach will control the boost by controlling the throttle and wastegate. Boost will then be related to torque to create a desired torque response.

SYSTEM OF INTEREST

In modern vehicles, the pedal position commanded by the driver is converted to a voltage that is sent to the engine control unit (ECU) and converted to a desired torque (i.e. the commanded/desired torque). Since torque in an actual vehicle cannot be directly measured without an expensive torque sensor, the desired torque can be related to manifold absolute pressure (MAP or boost), which is equivalent to the amount of air let into the engine measured in kPa.

In internal combustion engines, the throttle is a valve that regulates the air allowed into the engine, which is controlled with the gas pedal by the user. The air enters the cylinders where it is mixed with fuel and the combustion process occurs. This combustion creates power, which drives the pistons and moves the vehicle.

A turbocharged engine is a specific type of internal combustion engine that creates more power using recirculated exhaust gas. The wastegate is a valve that regulates how much exhaust gas is directed into the turbocharger to spin the turbine and how much leaves as exhaust from the vehicle.

These two components play the largest role in air flow and thus, the torque produced from the pedal command will be achieved by controlling these two actuators together. A basic diagram of the system with the mentioned actuators is shown in Figure 1 below.

The controller will relate the desired torque to a desired MAP through lookup tables, allowing for control variables that can realistically be measured in the real world and still ensure the goal of improved torque response is met.

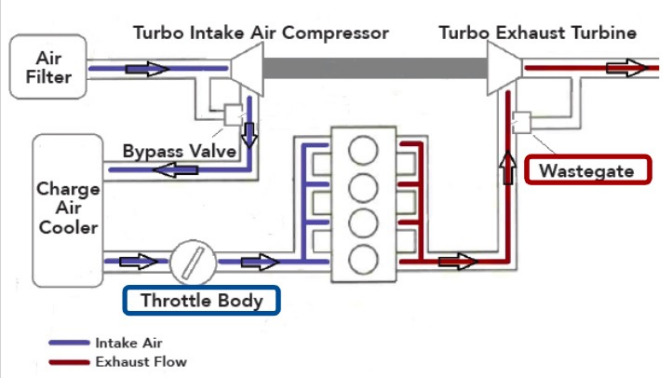

Figure 1: Airflow System In a Turbocharged Engine

MODELING AND SYSTEM IDENTIFICATION

In this project, the engine system will be simulated using the SI Engine Dynamometer in the MathWorks Powertrain Blockset, shown in Figure 2 below.

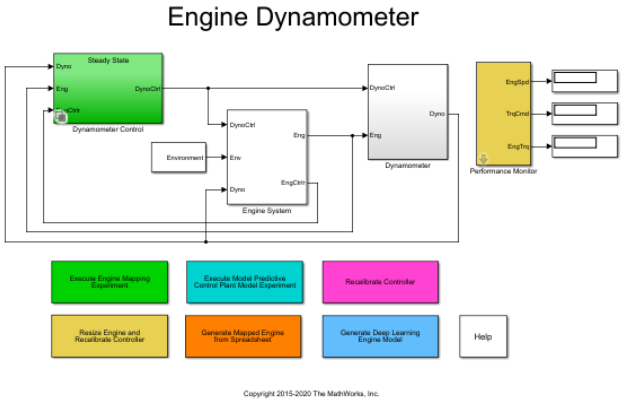

Figure 2: Engine Dynamometer Model

To model the system, the System Identification Toolbox in MATLAB was used to develop second order state-space models relating the throttle and wastegate to MAP separately using data acquired from the virtual engine simulation. Because the system is nonlinear, it can be linearized by creating models over a range of operating points (engine speeds and torque commands).

DATA COLLECTION In order to create the MPC controllers, each controller will need a model at specific operating points to reference while making control calculations. To do this, simulations were run to collect input/output data for a range of engine speeds; the data for engine speed, throttle position, waste gate position, and MAP needed to be collected for each simulation. This was done by using a ”To Workspace” block in Simulink to send the data to the MATLAB workspace for manipulation. Two sets of data were collected at each operating point/engine speed, one set for system identification and the other for system validation.

MODELING APPROACH The modeling was performed by overriding the engine speed to hold the engine in steady-state at a constant speed. The throttle and wastegate values in the controller were then overridden with a pseudo-random binary sequence (PRBS) sent into the throttle and wastegate values, essentially ’wiggling’ each actuator and then seeing how the intake manifold pressure (MAP) responded. When a PRBS signal was sent into the throttle, the wastegate position was held constant to see only the effect of the throttle and vice versa. In order to improve accuracy, we utilized a system called differential modeling, which detrended the data (’detrend’ command used for input/output data), then reduced the magnitude of the data to improve the system modeling. MPC works best in ”differential control mode.” Differential

modeling gets its name because it creates a model from the differences of the inputs/outputs; essentially the mean values of the inputs/outputs are are subtracted from the measured input/output data to see the direct changes in the output made by changes in the input. The process reduced the value of the manifold pressure so that the midpoint of the values was zero.

Modeling Throttle At small throttle angles the boost is sensitive, since a small change in its position would correlate to a large percent difference in air pressure in the manifold. MAP is also much easier to model and control at small throttle angles; as the percent position of the throttle increased, the system became much more difficult to model.

To model the relationship between throttle and MAP, the wastegate was held completely open at 100% (no effect) and the throttle was 'wiggled' at low values. Throttle values that provided suitable models were very low, less than ten percent. An image of the data collected for the variables MAP, throttle position, wastegate position, and engine speed at one operating point is provided below.

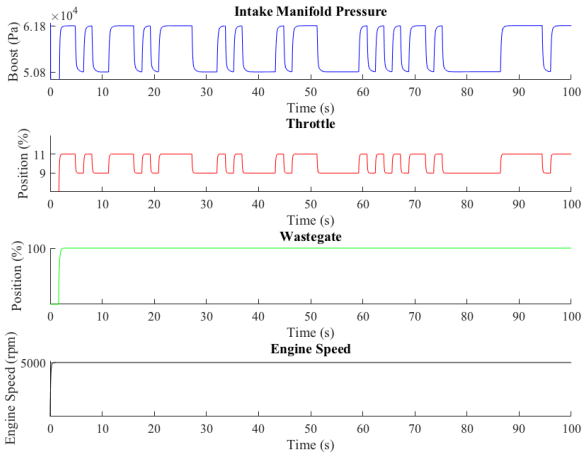

Figure 3: Throttle PRBS Signal and MAP Response at Engine Speed 5000

Modeling Wastegate The modeling of the wastegate was done in a similar manner to the throttle. Wastegate is much more sensitive when closer to closed (more air to spin turbocharger and thus, more air entering the cylinders), for the same reason throttle is sensitive at small angles. To model the wastegate, a PRBS signal was sent to override the wastegate, and the throttle was held completely open at 100%. This was done to ensure only the wastegate would have an impact on the MAP response, and create a wastegate-MAP model. An image of the data collected for the variables MAP, throttle position, wastegate position, and engine speed.

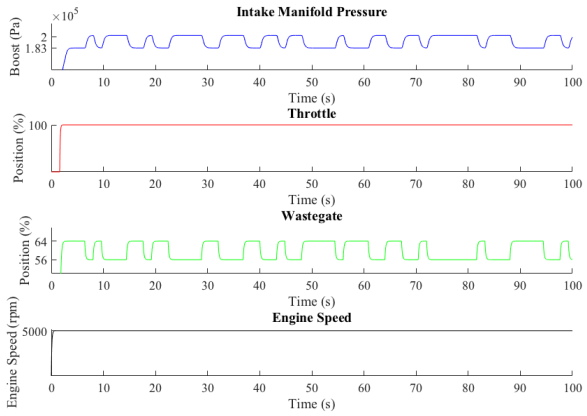

Figure 4: Wastegate PRBS Signal and MAP Response at Engine Speed 5000

SYSTEM IDENTIFICATION A black box modeling approach was used, putting the identification input and output data into MATLAB's System Identification Toolbox. This was automated with scripts and the use of the n4sid command. To relate each input variable to the intake manifold pressure, only 2nd order models were required to produce accurate models.

MODEL VALIDATION To validate the states-spaces created with the identification data set, and a second validation set of data was then collected at the same parameters (different PRBS signal). The inputs of the second data set were input into the state-space, creating a 'best guess' output from our system. This is referred to as the model or fitted output. The system output and the model output were then compared by plotting them against one-another. The MATLAB compare function was used to return a number fit by returning a normalized root mean square error (NRMSE) value, shown in Equation 1, that returned fits. The aim was to have fits above 90%.

$$fit = 100(1 - \frac{||y - \hat{y}||}{||y - \bar{y}||}) \quad (1)$$

Where y is the validation data output, $\hat{y}$ is the system output, and $\bar{y}$ is the mean of y. Below are the fits for the throttle (Figure 5) and the wastegate (Figure 6) at one operating point.

AUTOMATION

To speed up the process described above, we automated the collection and validation of data. Scripts were written in MATLAB to acquire the identification and validation data from the Powertrain Blockset; the scripts must be run with the modifications made in Simulink (see Operation Manual for changes), so that state-space models and MPC objects can be created for specific operating points defined by the user. This data was then pre-processed

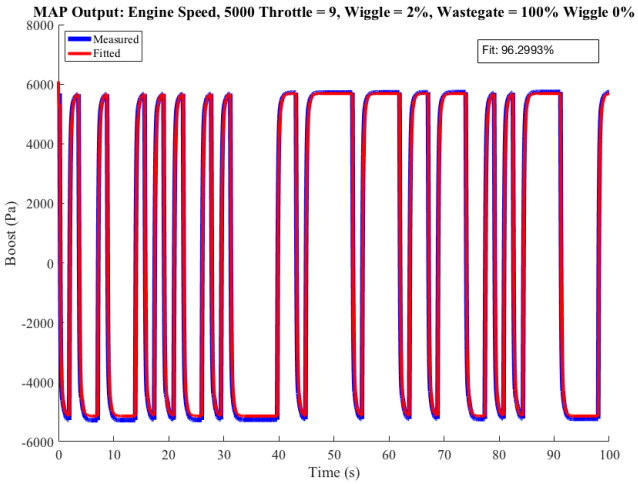

Figure 5: Fitted Model for Throttle over Measured Boost Data at Engine Speed 5000

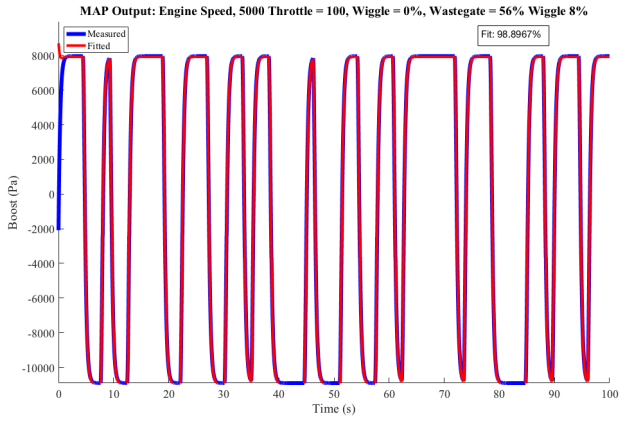

Figure 6: Fitted Model for Wastegate over Measured Boost Data at Engine Speed 5000

to produce more accurate results and state-space models that will work well when implemented into the multiple MPC controller blocks. Throttle and wastegate values that produce exceptional models for a range of operating points are shown in the tables below. This is done for a smaller number of operating points for the wastegate (since wastegate only begins making an impact around 2000 RPM) and the change in throttle position values remained relatively the same across the range.

Table 1: Throttle Values with Good Fits for Range of Operating Points

Eng Spd (rpm)	Avg MAP (kPa)	Thr Pos (%)	Wiggle
750	45920	0.8	0.25
1250	36230	1.0	0.5
1750	35340	1.5	0.55
1755	60250	2.8	1
2000	56860	3.0	1.2
3000	39030	3.0	1.5
3250	39390	3.2	1.9
3500	43850	4.2	1.9
3750	43580	4.6	1.9
3850	45900	5.1	1.9
3950	48120	5.6	1.9
4000	49860	6.0	1.8
4250	49780	6.4	1.9
4500	50670	7.1	1.8
4750	56170	8.7	1.6
5000	56400	9.0	2
5250	56060	9.3	2.2

Table 2: Wastegate Values with Good Fits for Range of Operating Points

Eng Spd (rpm)	Avg MAP (kPa)	Thr Pos (%)	Wiggle
2000	102850	62	10
3000	122780	56	9
4000	169690	57	9
5000	189930	55-58	8

RESULTS AND VALIDATION

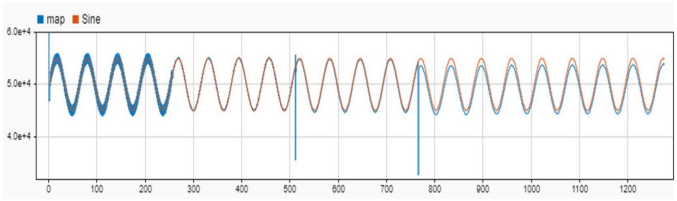

Figure 7: Throttle Control at Four Different Engine Speeds

The figure above is the engine's absolute manifold pressure (MAP) output (blue) that results from the throttle values controlled by the MPC, compared to the desired output (orange), which for this example was a sine wave. The state-space models used for these MPCS related throttle to MAP (note that it was not differential MAP). The operating points tested worked around four different engine speeds. From left to right on the plot, they are 750 RPM, 1000 RPM, 2500 RPM and 5000 RPM. Even with the noise in the output of the 750 RPM section, the error across the system is only 1%, far lower than our goal of 1.75%. As the output follows the desired MAP, there is lag error present, but it is minimal. In between operating points 1000 RPM, 2500 RPM, and 5000 RPM, a spike in MAP can be observed. This is due to the RPM suddenly

spiking and creating confusion for the MAP controller response. The MPCs controlling the throttle aren't able to respond instantly, causing a very brief error as the throttle is adjusted. The error has little impact on the accuracy of our controller, and can effectively be ignored.

To implement differential boost control models into an MPC controller for testing, the mean boost value was subtracted from the boost command before entering the MPC, and the nominal actuator values were added to the resulting control moves for differential boost leaving the MPC to return the actual actuator positions required.

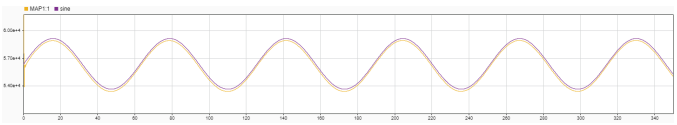

Figure 8: Steady-State Relative Throttle Control of Differential MAP Engine Speed 5000

The figure above shows relative throttle control; this uses the state-space model created with the differential MAP values. As explained previously, the midpoint value was then added after the control decisions were made by the MPC. When we attempted the same solution for wastegate control, the accuracy of our systems were unusable. Regardless of whether the throttle or wastegate was being tested, the control method was the same.

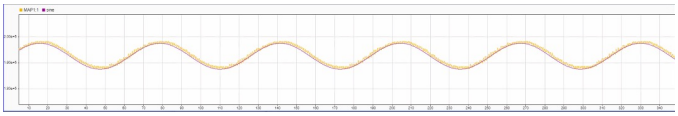

Figure 9: Steady-State Relative Wastegate Control of Differential MAP Engine Speed 5000

Figures 8 and 9 show the results of our differential models. While we have only been able to get one state-space working for the wastegate-MAP relationship so far, it shows promise. Figure 8 shows the results of the differential model working with throttle, and has an error of about 0.5%. Figure 9 shows the results of the wastegate and boasts an error of 0.4%.

Differential modeling, while it produced better fits, was found to limit the operating point range for the output response. This means the models require more operating points to output the desired boost response.

HARDWARE-IN-THE-LOOP To ensure this controller can be run on a real-world electronic control unit (ECU), we created a hardware-in-the-loop (HIL) system to run alongside the simulation. Figure 10 shows a picture of our final HIL system.

The HIL system receives the simulated throttle and wastegate positions created by the MPC, using an Arduino Uno as the data acquisition and motor driver. The Arduino then commands a servo motor to move to the simulated angle, and an encoder measures that angle digitally. The protractor shown on the right of the image above will be used to visually check the throttle/wastegate angle. After the encoder measures the angle, the digital signal is sent back to MATLAB to override the simulated throttle/wastegate valve angle. If the MAP output is as expected, the HIL system has verified the control method can be applied to a real-world ECU.

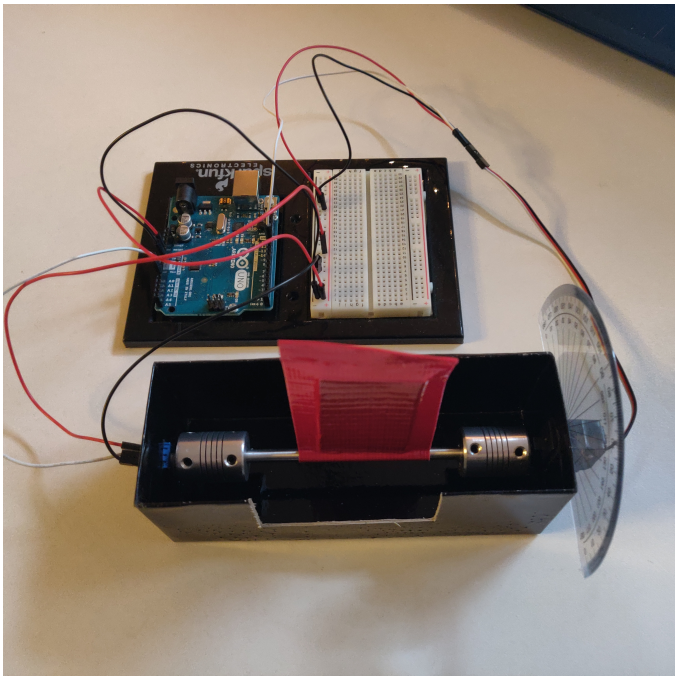

Figure 10: Hardware-in-the-Loop System

FUTURE WORK

We successfully created a system that accurately switches between MPCs under different operating ranges to control throttle position to produce a desired MAP command. We were also able to create state-spaces that utilize differential MAP, throttle, and wastegate that were able to return actuator values to achieve the desired MAP command in steady state; this testing was done for one operating point at a time. The next steps in this project would be creating an MPC system to coordinate the control of the two variables, throttle and wastegate positions, in transient operation to achieve the desired torque command. This can be done with a lookup table referenced by the multiple MPC block that switches between MPC controls as needed. In steady-state conditions, the current controller will use the lookup tables already in place to make control decisions. As the system response reaches transient state between those steady-state points, the MPC control will be turned on and help control the throttle and wastegate positions to reach the next steady-state condition.

The creation and implementation of two-dimensional lookup tables that reference both desired boost and current RPM would allow a scheduled set of MPCs to be ref-

erenced by the controller. Creating more models across the range of operating points would allow the multiple MPC block to have a more refined control of the airflow, which would create more accurate responses. This comes at the price of heavier computing power, so it is important to find a balance between reduced error and number of models.

Additionally, work would need to be done to test for measured disturbances (changes in engine speed) and unmeasured disturbances (manifold leak, throttle body build tolerance issues, etc.) to be included into the model predictive controllers.

CONCLUSION

The ultimate goal of this project was to achieve transient and steady state torque control in the MathWorks SI Engine Dynamometer in the Powertrain Blockset application using a multi-input multi-output (MIMO) approach. Input/output data was generated using the SI Engine Dynamometer. Modifications were made to the Dynamometer (outlined in the Operation Manual), and scripts were written to automate the creation of state-space models using MathWorks' System Identification Toolbox. MPC objects were created based on those models through the scripts for quick and easy implementation. Accurate second order models relating each actuator to differential MAP were created, and were used to create control predictions by the MPC controller.

It is important to note that the original goal of a multi-input multi-output system was not achieved. The implemented controller can control throttle and wastegate individually, but not together to output a single matrix of improved throttle and wastegate positions. While we did not achieve a MIMO system, we established an excellent framework to continue expanding on the MPC control of this system, and are confident our work can be used as stepping stones for the continuation of this research.

REFERENCES

[1] Bemporad, A., Bernardini, D., Long, R., and Verdejo, J., "Model Predictive Control of Turbocharged Gasoline Engines for Mass Production," SAE Technical Paper 2018-01-0875, 2018, doi:10.4271/2018-01-0875.

AUTHOR CONTACT INFORMATION

Austin LaFever: all15g@my.fsu.edu
Patrick Marlatt: patrick1.marlatt@famu.edu
Frederick Peterson: fep17@my.fsu.edu
Jonathan Wozny: jw17m@my.fsu.edu

ACKNOWLEDGEMENTS

We would like to thank our project advisors, Peter Maloney and Roberto Valenti of MathWorks. Their insight helped us accomplish what we have, and we could not have completed this project without them. Thank you both!

We would also like to thank the FAMU-FSU College of Engineering for hosting MathWorks as a client and allowing us to learn through the application of the engineering process.

ABBREVIATIONS

1. MPC - Model Predictive Control, the control method used for this project.
2. MIMO - Multiple-Input Multiple-Output, the control approach for our MPC.
3. SISO - Single-Input Single-Output, the current transient control approach before implementation of our controller.
4. MAP - Intake Manifold Absolute Pressure, the measurable variable that relates to torque for control of airflow.
5. NRMSE - Normalized Root Mean Squared Error, the mathematical method to detrend and control the differential MAP.
6. PRBS - Pseudo-Random Binary Sequence, the signal sent into throttle or wastegate to override the initial value and create a throttle/wastegate-MAP relationship.
7. RPM - Revolutions Per Minute, the variable that defines the rotational speed of the engine.
8. ECU - Electronic Control Unit, the computer in a vehicle that makes control calculations for variables to be manipulated.
9. HIL - Hardware-in-the-Loop, the validation method used to ensure our controller can be applied to a real-world ECU.