Fault Tolerant Model Predictive Control of Quad-Rotor Helicopters with Actuator Fault Estimation

Hojjat A. Izadi*, Youmin Zhang*, and Brandon W. Gordon*

* Department of Mechanical and Industrial Engineering, Concordia University, Montreal, Canada (e-mails: {ha_izadi, ymzhang, bwgordon}@encs.concordia.ca)

Abstract: Model predictive control (MPC) at each time step minimizes a cost function subject to dynamical constraints to obtain a stabilizing control signal. Further, MPC is one of the few methodologies that can be used to design feedback control for nonlinear dynamical systems taking into consideration of actuator saturations. It can thus serve as a suitable fault tolerant control approach for quad-rotor helicopter governed by nonlinear dynamics. However, MPC needs a relatively accurate model of the post-failure system to calculate a stabilizing control signal. The problem becomes more critical where the system dynamics is described by a nonlinear model, because there exist few effective nonlinear parameter estimators with reasonable online computation time. To address this issue, for online actuator fault estimation, this paper investigates Moving Horizon Estimation (MHE) and Unscented Kalman Filter (UKF) as two methods for nonlinear parameter estimation. A framework is then formulated for integrating MHE/UKF based fault estimator with MPC to form an active fault tolerant control system for systems with nonlinear constrained dynamics. Performance and computation requirement of both algorithms are also investigated.

Keywords: Fault Tolerant Control, Model Predictive Control, Nonlinear Parameter Estimation, Quad-rotor Helicopter, Unscented Kalman Filter, Moving Horizon Estimation.

1. INTRODUCTION

Model predictive control (MPC), due to its prominent capabilities such as constraint handling, flexibility to changes in the process dynamics, and applicability to nonlinear dynamics, is potentially a promising tool for fault tolerant control applications (Maciejowski & Jones (2003), Maciejowski (1998), Zhang & Jiang (2008), Deshpande, et al. (2009) and Hennig & Balas (2008). Since MPC recalculate the control signal at each sampling time, any change in the process dynamics can be reflected simply into the control signal calculation. Also, the constraint handling capability of MPC allows system working close to the boundaries of the tight post-failure operation envelope. In addition, MPC performs control reallocation/redistribution while satisfying constraints for redundant dynamics such as quad-rotor helicopter. However, similar to most of the control techniques, MPC needs an almost explicit model of the system to calculate a stabilizing control signal. On the other hand, the abrupt changes in the model parameters, due to failure, cannot be predicted beforehand and an online data-driven parameter estimation methodology is required to extract the post-failure model parameters from online input/output data.

Due to the fact that the post-failure recovery time is very critical to rescue the faulty system, a sufficiently fast fault parameter estimation methodology is required. However, the existing nonlinear parameter estimators are known to be computationally time consuming. Therefore, the objective of this paper is to investigate the performance and computational load of Moving Horizon Estimation (MHE) and Unscented Kalman Filter (UKF) methods for actuator fault parameters estimation, in junction with MPC-based fault tolerant control to form an active fault tolerant control system. Both methods are known to provide promising results for state and parameter estimation of highly nonlinear systems. However, the computational load of both algorithms still remains as a concern for fast online processes. The most time consuming part of UKF is the square root calculation of some state covariance matrix at each time step. The high computation time of MHE comes from the fact that the state/parameter is calculated by minimizing a cost function over a moving estimation horizon at each time step. Reducing the estimation horizon can help to reduce the online computation time of the MHE. However, stability becomes a barrier. The number of parameters to be estimated can also affect the online computation time.

![Fig. 1. Fault tolerant MPC with parameter estimation](image-url)
the fault for tandem steel mill. Also, in Samar et al. (2006), MHE is used to estimate the fault parameters of an unmanned aircraft. Fig. 1 shows a schematic of a typical fault tolerant MPC architecture equipped with a fault parameter estimator. Parameter estimation block, in this paper, uses MHE or UKF to estimate the fault parameters. MPC uses parameterized model to calculate the control signal. Then at any time step, by using the input and output information of the process, the parameter estimation block calculates the fault parameters to be used by MPC in on-line.

2. QUAD-ROTOR HELICOPTER DYNAMICS

Fig. 2 shows the configuration of a quad-rotor helicopter in xyz coordinate. Many papers have presented the dynamics of quad-rotor with the configuration shown in Fig. 2, for example see Lee et al. (2009). Eq. (1) represents the quad-rotor dynamics in hover maneuver with damping terms:

\[
\begin{align*}
\dot{x} &= u_x (\cos \phi \sin \theta \cos \psi + \sin \phi \sin \psi) - k_x \dot{x} / m \\
\dot{y} &= u_y (\cos \phi \sin \theta \sin \psi - \sin \phi \cos \psi) - k_y \dot{y} / m \\
\dot{z} &= u_z \cos \phi \cos \theta - g - k_z \dot{z} / m \\
\theta &= u_\theta - L_k \phi \theta / J_\theta \\
\phi &= u_\phi - L_k \phi \phi / J_\phi \\
\psi &= u_\psi - L_k \psi \psi / J_\psi
\end{align*}
\]

(1)

where \( x, y \) and \( z \) represents the position vector components, \( \theta, \phi \) and \( \psi \) denote pitch, bank and yaw angle respectively. Also, \( J_\theta = J_\phi = J_\psi = 0.03 \text{ kg.m}^2 \), \( m = 1.4 \text{ kg} \), and \( L = 0.2 \text{ m} \) represent the moments of inertia, mass and distance of motors from center respectively. \( g = 9.81 \text{ m/s}^2 \) represents the gravity acceleration. \( k_x, k_y, k_z, k_\theta, k_\phi \) and \( k_\psi \) are the drag coefficients which are negligible. Then, the control inputs are given as follows:

\[
\begin{align*}
u_x &= (F_1 + F_2 + F_3 + F_4) / m \\
u_\theta &= L(F_1 - F_2) / J_\theta \\
u_\phi &= L(F_2 - F_4) / J_\phi \\
u_\psi &= c(F_1 + F_3 - F_2 - F_4) / J_\psi
\end{align*}
\]

(2)

where \( F_1, F_2, F_3 \) and \( F_4 \) are forces generated by four rotors connected separately to four DC motors. Also, the inputs are constrained as follows:

\[
0 \leq F_i \leq 8 \text{ (N) } \quad i = 1, 2, 3, 4
\]

(3)

As shown by simulation in Section 5, MPC performs necessary control redistribution in the presence of partial actuator (rotor) fault and provides acceptable performance under fault conditions.

3. MODEL PREDICTIVE CONTROL (MPC)

3.1. MPC Notation and Terminology

With the model predictive control (MPC), also known as receding horizon control (RHC), a cost function is minimized over a future prediction horizon time step denoted by \( N \), subject to the dynamical constraints. The first control input in the sequence is applied to the plant until the next update is available. The reader is referred to Mayne et al. (2000) for a comprehensive review of the existing MPC schemes.

The discrete timing is shown by \( k \) where \( k \in \mathbb{N} \). The possible state vectors are introduced as follows:

\[-x(k) \text{ : the actual state vector at time step } k.\]

\[-x_\phi(k) \text{ : the predicted state vector at time step } k+j,\text{ computed at time step } k, \text{ where } j \in \{0,1,2,...,N\} \text{.}\]

Similar notation is used for the control input vector \( u \). Also the sequence of the state/input vector over the prediction horizon is called the state/input trajectory and is shown as follows:

\[-x_\phi(k) = \{ x_\phi(k+j) | j = 0,1,2,...,N \}\]

\[-u_\phi(k) = \{ u_\phi(k+j) | j = 0,1,2,...,N-1 \}\]

(4)

3.2. MPC Formulation

The cost function at time step \( k \) is defined as follows:

\[J(x_\phi(k),u_\phi(k)) = \sum_{j=0}^{N-1} \left( \left\| x_\phi(j) - r \right\|_Q^2 + \left\| u_\phi(j) \right\|_R^2 \right) + \left\| x_\phi(N) - r \right\|_P^2\]

(5)

where \( \left\| x \right\|_Q^2 = x^T Q x, P > 0, Q > 0, \text{ and } R > 0 \) are symmetric matrices. Also, \( r \) is the state vector of target state (or reference state).

Then, MPC problem \( \mathcal{P}_k \) at time step \( k \) is defined as follows:

**Problem 1: MPC Problem \( \mathcal{P}_k \) :**

Calculate:

\[J^*(x(k)) = \min_{\{u_\phi(k),x_\phi(k)\}} J(x_\phi(k),u_\phi(k))\]

(6)

subject to (for \( j = 0,1,2,...,N-1 \)):

\[x_\phi(j+1) = f(x_\phi(j),u_\phi(j)) \quad ; \quad x_\phi(0) = x(k)\]

(7a)

\[x_\phi(j) \in \mathbb{X} \& u_\phi(j) \in \mathbb{U}\]

(7b)

\[x_\phi(N) \in \mathbb{X}_f\]

(7c)

where \( \mathbb{X} \subseteq \mathbb{R}^m \), \( \mathbb{U} \subseteq \mathbb{R}^m \) and \( \mathbb{X}_f \subseteq \mathbb{X} \) denote the set of admissible states, inputs and terminal states (terminal region) respectively. Superscript * denotes the optimal value of the parameter. MPC formulated here is based on the quasi-infinite model predictive control in Chen & Allgower (1998).

3.3. MPC Algorithm

At each time step \( k \), MPC generates the input and state trajectories, by solving the optimization problem \( \mathcal{P}_k \). After generating these trajectories MPC controller applies only the
first computed control input, i.e. \( u_k(0) \) to the system. The following algorithm presents the online implementation of MPC:

**Algorithm 1: MPC**

Given \( x(0) \) and \( r' \), do:
1. \( k = 0 \).
2. Measure (or estimate) \( x(k) \).
3. Solve \( \mathcal{F}_k \) and generate \( u_k(\cdot) \) and \( x_k(\cdot) \).
4. Apply \( u_k(0) \) to the system.
5. \( k = k+1 \) and GOTO step 2.

This algorithm is repeated for \( k = 0,1,2,...,\infty \). In step 2, if full state measurement is not available, then the state estimation is performed using UKF or MHE, to be described in Section 4.

4. Fault Tolerant MPC with Fault Estimation

To model the actuator fault, the system dynamics is reformulated as:

\[
x(k+1) = f(x(k), \alpha(k)u(k))
\]

where \( \alpha \) captures the fault information and is called the fault parameter matrix which determines the fault severity. \( \alpha \) is a diagonal matrix, i.e.

\[
\alpha = \text{diag}(\alpha_1, \alpha_2, ..., \alpha_m)
\]

where \( m \) is the number of inputs and the scalars \( \alpha \) denotes the fault estimation on actuator \( i \). For a healthy system \( \alpha = I \) and for complete loss of actuators effectiveness \( \alpha = 0 \). For partial loss of actuator \( i \): \( 0 < \alpha_i < 1 \). The objective of the fault identification is the online estimation of the fault parameter matrix \( \alpha \), using the available post-failure system input/output (I/O) data by MHE or UKF.

4.1. Fault Identification using MHE

Moving horizon estimation (MHE) (Rao et al., 2003) uses a series of past measurements over the finite previous time window, called estimation horizon, to calculate the states or system parameters over the estimation horizon. MHE minimizes a cost function called identification cost, as follows:

\[
J^{ID}(\Gamma_k, \alpha_k(\cdot)) = \sum_{p=0}^{N_f-1} \left\| x(p+1) - f(x(p), \alpha_k(p)u(p)) \right\|^2_S
\]

where \( S > 0 \) is symmetric matrices and \( \Gamma_k \) denotes the set of previous I/O data of the process:

\[
\Gamma_k = \{ (x(k-p), u(k-p)) \mid p = 1,2,...,N_f \}
\]

where \( N_f \) is called estimation horizon (the number of I/O data used for model identification). Also, the sequence of the fault parameters is defined as follows:

\[
\alpha_k(\cdot) = \{ \alpha_k(j) \mid j = k-N_f, k-1 \}
\]

**Remark 1:** MHE is often used for state estimation from output measurements. The general formulation can be found in Rao et al. (2003) and Rao & Rawlings (2002). In this paper, since we are interested in the parameter estimation capability of MHE, the presented formulation may differ from the general formulations of MHE. The MHE formulation is intended to be formulated such that the online computation time is reduced. Then the state estimation capability is removed and the cost function is only a function of fault parameters and I/O information.

Given the I/O data set \( \Gamma_k \), through minimizing the identification cost \( J^{ID}(\Gamma_k, \alpha_k(\cdot)) \), \( \alpha_k(\cdot) \) is determined. Then the parameter estimation is formulated as below:

**Problem 2: Moving Horizon Estimation (MHE) Problem:**

Calculate:

\[
J^{ID*}(\Gamma_k) = \min_{\alpha_k(k-1)} J^{ID}(\Gamma_k, \alpha_k(\cdot))
\]

subject to (for \( j = k-N_f, ..., k-1 \)):

\[
x(j+1) = f(x(j), \alpha_k(j)u(j))
\]

and

\[
0 \leq \alpha_k(j) \leq I
\]

Although the operation \( \leq \) is not defined for matrices, we abuse \( \leq \) in (15) to simplify formulation. \( \leq \) is applied to the corresponding entry of the compared matrices. Then the solution of the above optimization problem is:

\[
\alpha = \alpha_k(k-1)
\]

4.2. Fault Tolerant MPC

The fault tolerant MPC problem is defined as follows:

**Problem 3: Fault Tolerant MPC Problem** \( \mathcal{F}_k^F \):

Calculate:

\[
J^*(x(k)) = \min_{\{u_k(\cdot), x_k(\cdot)\}} J(x_k(\cdot), u_k(\cdot))
\]

subject to

\[
x_k(j+1) = f(x_k(j), \alpha u_k(j)) ; \quad x_k(0) = x(k)
\]

and

\[
(7b) \quad \& \quad (7c)
\]

The following algorithm is presented for the online implementation of the proposed fault tolerant MPC problem \( \mathcal{F}_k^F \).

**Algorithm 2: Fault Tolerant MPC with parameter estimation:**

Given \( x(0) \) and \( r' \), do:
1. \( k=0 \).
2. Measure \( x(k) \).
3. Update \( \Gamma_k \).
4. Solve Problem 2, to calculate \( \alpha = \alpha_k(k-1) \).
5. Solve \( \mathcal{F}_k^F \) and generate \( u_k(\cdot) \) and \( x_k(\cdot) \).
6. Apply \( u_k(0) \) to the system.
7. \( k=k+1 \) and GOTO step 2.

This algorithm is repeated for \( k = 0,1,2,...,\infty \). Note that at each time step \( \Gamma_k \) is updated with new information.
4.3. Fault Estimation using Unscented Kalman Filter (UKF)

The Extended Kalman Filter (EKF) linearizes the nonlinear dynamics by calculating the Jacobian at each time step. While computation of Jacobian is time-consuming, it was applicable to systems with smooth and at least once differentiable dynamics. Recently, the Unscented Kalman Filter (UKF) estimators have been used which do not need the calculation of Jacobian at any step. Rather, it generates some statistical sigma points around the measurements and estimates the states by maximizing some probability indices. The most time consuming part of UKF is the square root calculation of state covariance at each time step to generate sigma points. UKF is applicable to a more general class of nonlinear dynamics and it offers enough stability proofs. A detailed formulation of UKF can be found, for instance, in Qi & Han (2008), Ma & Zhang (2010). To avoid repeating the heavy standard formulation of UKF, they are not presented here.

Then, the UKF estimator can be used, instead of Problem 2 in the 4th step of Algorithm 2, for parameter estimation.

4.4. Fault Detection

Whenever the estimated system parameters, e.g. actuator effectiveness, deviate from their nominal value (\( \alpha = I \)) a fault is concluded.

5. SIMULATION RESULTS

The proposed fault tolerant Algorithm 2 is applied to the hover control of quad-rotor helicopter. Due to the available actuation redundancy in terms of the number of rotors compared with single-rotor helicopter, it is a suitable test-bed to evaluate the fault tolerant control schemes.

In all cases the quad-rotor is on the ground initially and it is desired that it reaches a hover height of 4 m and stay in that height while stabilizing the pitch and bank angle. Other states are desired to be:

\[
r = [2,0,3,0,4,0,0,0,0,0,0,0]
\]

Fig. 3 shows the time history of states for fault-free condition (the desired position values are dotted and the velocities are dashed lines). In this case Algorithm 1 is used, where \( \alpha = I \) (or \( \alpha_1 = \alpha_2 = \alpha_3 = \alpha_4 = 1 \)).

Fig. 4 shows the time history of the states when no fault estimation is performed. In fact in this case Algorithm 1 is used. Fig. 4 shows that states do not converge to their desired values and therefore it exhibits poor performance. Fig. 4 shows also that interestingly MPC exhibits some degree of fault tolerance inherently; all the linear and velocities are stabilized and there is only some error in the positions and orientations. In the next sections, UKF and MHE fault estimators are used to recover the faulty quad-rotor helicopter.

5.1. Fault Injection

The fault is designed to happen at time \( t_f = 5 \) sec. At this time it is assumed that actuator failures occur which lead to multiple simultaneous partial loss of effectiveness of three actuators as follows:

\[
\alpha_1 = 0.9; \alpha_2 = 0.7; \alpha_3 = 0.8; \alpha_4 = 1.0
\]  

(18)

Such kind of multiple simultaneous faults may happen due to severe weather conditions damaging the rotors or physical collision with obstacles.

Fig. 4 shows the time history of the states when no fault estimation is performed. In fact in this case Algorithm 1 is used. Fig. 4 shows that states do not converge to their desired values and therefore it exhibits poor performance. Fig. 4 shows also that interestingly MPC exhibits some degree of fault tolerance inherently; all the linear and velocities are stabilized and there is only some error in the positions and orientations. In the next sections, UKF and MHE fault estimators are used to recover the faulty quad-rotor helicopter.

5.2. Fault Estimation with Unscented Kalman Filter (UKF)

Fig. 5 through Fig. 7 show the simulation results for the faulty situation when 3 motors lose some portion of their effectiveness as defined in (18), and also Algorithm 2 is used with UKF estimator (at step 4) for fault parameter estimation. Fig. 5 shows that the states converge to their desired value gradually. Fig. 6 shows that the time history of the estimated parameters and their desired value. It takes about 12 sec after the fault for parameters to converge to their actual values. The corresponding input profile is also shown in Fig. 7, which shows that the control inputs satisfy the saturation constraint. It also shows how the controls are redistributed after occurrence of the fault.

5.3. Fault Estimation with Moving Horizon Estimation (MHE)

Algorithm 2 is used now with MHE as fault parameter estimator. The estimation horizon for this case is chosen to be \( N_I = 3 \). Fig. 8 through Fig. 10 show the simulation results for the case when fault parameters are estimated using MHE (Problem 2 in step 4 of Algorithm 2). Comparing Fig. 8 and
Fig. 5. States for faulty condition with UKF fault estimation

Fig. 6. Fault parameters for faulty condition with UKF fault estimation (solid lines represent the estimated values and the dotted lines represent the actual value of the parameter)

Fig. 7. Inputs for faulty condition with UKF fault estimation

Fig. 8. States for faulty condition with MHE fault estimation

Fig. 9. Fault parameters for faulty condition with MHE fault estimation (solid lines represent the estimated values and the dotted lines represent the actual value of the parameter)

Fig. 9 with Fig. 5 and Fig. 6 respectively, slightly improved performance of MHE estimator comparing with the UKF estimator can be viewed. Fig. 9 shows that the parameters converge to their actual values immediately after fault occurrence, benefited from the moving horizon estimation. This improvement is achieved due to the fact that UKF assumes no dynamics for fault parameters. However, MHE considers a time-varying nature for fault parameters and tries to calculate the implicit dynamics by optimization and then matching the I/O information.

The input profile is depicted in Fig. 10 (comparing with Fig. 7). Again it shows that the control inputs satisfy the saturation constraint. Also, MPC redistributes the remaining control inputs after the fault effectively. Note that all the simulations are performed under the same conditions as in Section 5.2 for UKF.
5.4. Comparison of performance and computation load between UKF and MHE

Following analysis implies that UKF require a slightly lower computation time. For three different faulty conditions with the overall scenario explained before, the computation time is measured as follows:

\[
T = \frac{\text{Total Measured CPU Time}}{\text{Total Simulation Time}}
\]  \hspace{1cm} (19)

For the controller to be implementable in real-time it is required that \( T < 1 \). The results are gathered in table below:

<table>
<thead>
<tr>
<th>Case</th>
<th>T (UKF)</th>
<th>T (MHE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.1735</td>
<td>1.222</td>
</tr>
<tr>
<td>2</td>
<td>1.3983</td>
<td>1.500</td>
</tr>
<tr>
<td>3</td>
<td>1.903</td>
<td>2.15</td>
</tr>
<tr>
<td>4</td>
<td>0.954</td>
<td>0.9775</td>
</tr>
</tbody>
</table>

where, Case 1 - Case 4 differ only in the MPC parameters such as prediction horizon. Table 1 shows that, depending on the problem size the computation time varies but for all cases the computation time of UKF is slightly less than that of MHE.

6. CONCLUSIONS

A MPC (model predictive control) based fault tolerant controller has been integrated with a Moving Horizon Estimation (MHE) and/or Unscented Kalman Filter (UKF) for fault parameters estimation to form an active fault tolerant control system in this paper. Simulation results show that MPC-based fault tolerant controller provides prominent fault tolerant capabilities to the control of quad-rotor helicopter with constrained nonlinear dynamics. MPC is flexible to abrupt changes in the dynamics. Further, it satisfies all the system constraints and performs control redistribution effectively in an optimal manner. It is also seen that the fault parameters converge faster using MHE when compared with the UKF-based fault estimator. However, the computation time of MHE is slightly higher than that of UKF.

REFERENCES


