Models for Optimal Dynamic Reconfiguration and Simulation of Ship Power Systems in SIMULINK with Stateflow

Edoe Mensah Harry Kwatny[1] Dagmar Niebur
Jean-Etienne Dongmo
Drexel University

Gaurav Bajpai Carole Teolis
Techno-Sciences, Inc.

Abstract: The construction of models for power systems that allows for the simulation and analysis of power system involving discrete events is described. The models are useful for simulation as well as design of optimal power management systems that involve system reconfiguration and load shedding.

I. Introduction

The capability to dynamically reconfigure future naval integrated electric power system is central to the Navy's vision of the future combat ships. The objective of our project is to design, implement and evaluate (in a real time simulation environment) a Shipboard Power System Management system that will prevent loss of power at critical buses when damage conditions are encountered. This system will include intelligent discrete actions, such as load shedding, and will provide voltage security during battle conditions. The approach that we propose will optimally shed load, activate an uninterruptible power supply following the occurrence of disruptive events, switch vital load feeder and shift power supply distribution between generators as necessary.

Our approach is based on a new paradigm for the design of optimal control systems for *hybrid systems*, i.e., systems composed of continuous dynamics and discrete events. Discrete events may involve external disturbances, the discrete action of protection devices or control systems. The essence of the idea is that discrete acting subsystems are naturally associated with a set of logical conditions or *logical specifications* and the continuous system dynamics are usually described by differential or differential-algebraic equations. The key in our approach is to symbolically transform the logical specification that describes the discrete subsystem to a set of inequalities in integer-valued variables. These set of inequalities are called integer programming formulas, or simply *IP formulas*. The logic to IP package that we have developed in *Mathematica* is very general and comprehensive [1]. We devised a dynamic programming algorithm tailored to hybrid systems that solves dynamic optimization problems involving both binary (logical) and real variables [2].

We have established the feasibility of our approach in a 3-bus power systems example equipped with an Uninterruptible Power Supply (UPS). In this paper we consider a full version of the Integrated Powers System.

Our model of the Integrated Power System is a modification of the model described in [3, 4]. The system is composed of two generators feeding two induction motors through transmission lines. Vital loads as well as non-vital loads are connected to the various buses. The vital loads can be supplied

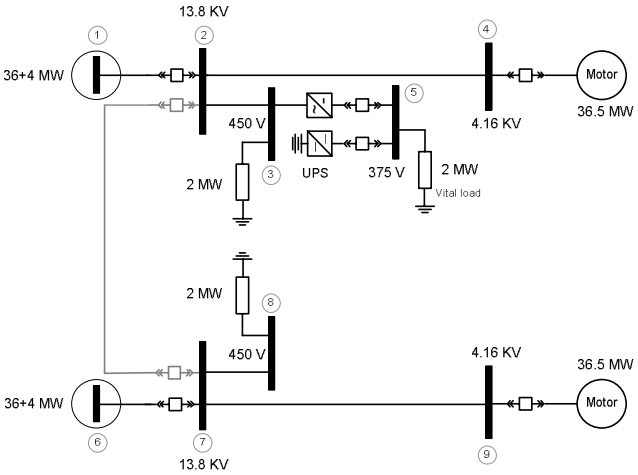

Figure 1 Notional DD(X) distribution system abstraction.

from either the port or the starboard side. The model is equipped with a UPS that can supply power to the vital loads when needed. One abstracted configuration of the network is shown in Figure 1.

II. Problem Definition

In this section we define the objectives of a power management system (PMS) in the context of a Navy all electric integrated power system. Typical PMS functionality includes

1. Restart from blackout – A blackout is the complete disconnection of all generators

[1] Corresponding author:: Harry Kwatny, MEM Department, Drexel University, 32nd & Chesnut Streets, Philadelphia, PA 19104. hkwatny@coe.drexel.edu.

2. Operational mode – Each operational mode has specific minimal requirements
3. Generation Start/Stop
4. Propulsion commanded power changes
5. Load shedding

Various faults scenarios are examined

- Line faults, characterized by full or partial reduction of the associated admittance.
- Loss of a single generator.

A. Reconfiguration Strategies:

Discrete states corresponding to admissible reconfigurations can now be defined and system models developed for each discrete state. In addition, a transition structure can be defined that expresses allowable transitions between the discrete states. While it is always possible to allow transitions from every discrete state to every other discrete state, the specification of a transition structure has many benefits. The specification allows us to impose constraints on the reconfiguration process and to eliminate unsuitable transitions at the outset.

The severity of the failure obviously requires that both motors be dropped to 50% or that motor 1 is dropped completely. It is preferable that both motors remain in operation so the first transition is clearly to a state with both motors at 50%. Figure 2 shows one possible transition specification.

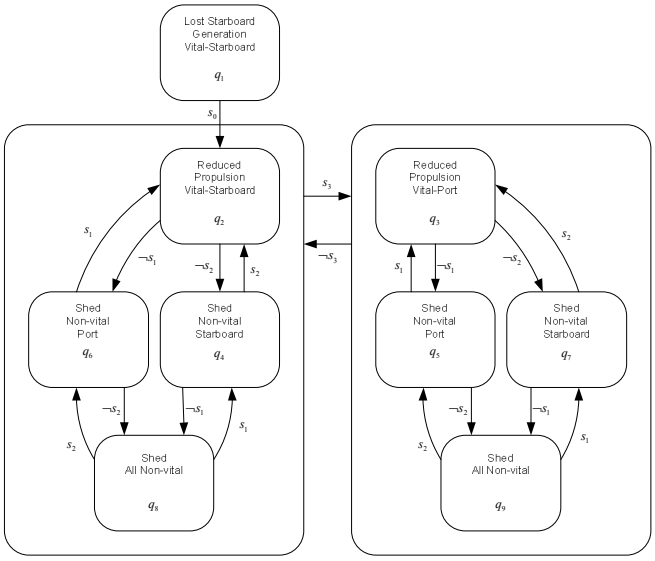

Figure 2. Discrete state transition diagram illustrates admissible discrete actions. Note the use of 'super state' to simplify diagram.

III. Simulation Approach

The dynamics of the generator and motor are written in terms of a differential-algebraic equation. Dynamics are also included for the battery model. These are symbolically discretized as we will describe below. A hybrid system of the integrated power systems with a UPS has been formulated. The system can be assembled with the *Mathematica* package *ProPac* as a SIMULINK model ready for simulation in MATLAB/SIMULINK/Stateflow.

In our concept of a modeling and design toolbox, we need models for control design, simulation, and for analysis, such as load flow or voltage stability. The models are associated with a common symbolic representation. These models can be assembled in a block diagram environment using standard components selected from a library. Each component will have at least two forms; a symbolic model and a SIMULINK *S-Function*, automatically created from the symbolic model. Our *ProPac* software has functions that create optimized C-code *mex* functions that compile into the required *S-functions*. Currently, we have classical network models (with any number of PV, PQ and generator buses) and various types of machine models.

The simulation block diagrams comprise a control panel and a state feedback controller connecting the power plant. The state feedback controller strategy for the mode switching of the power system is obtained off-line through Mixed Integer Dynamic Programming. It is computed in form of a lookup table that presents a mapping from combinations of predefined modes, events, generator angle, slip conditions and battery state to required switching actions. Scopes are provided for viewing the performance variables such as the internal voltage E, regulated voltage V, the state of charge of the battery σ as well as mode switching. Dividing the simulation in this manner allows the user complete control over the system behavior independent of the controller. It allows the inclusion of multi mode behavior and one sided limits as necessary. Additionally, this presentation facilitates the hardware-in-the-loop verification experimentation using SIMULINK's Real-time Workshop. We are currently implementing the software tools to translate the hybrid controller as a block in SIMULINK. Further simulation results will be established once the optimal state feedback switching control strategy is available as a lookup table.

IV. Dynamics Described by Differential-Algebraic Equations

Differential-Algebraic-Equations (DAEs) [5] form the essential mathematical model for power systems. So it is not surprising that a great deal of attention has been paid to the development of computational methods for solving them. Nevertheless, computing trajectories remains problematic and analysts often need to experiment with a variety of methods and parameters before obtaining satisfactory results. When the system involves switching or mode transitions the difficulty is magnified many times, and, to this date, very little thought has been given to hybrid systems with continuous dynamics described by DAEs.

In previous work, specifically [2], we were able to solve the algebraic equations (the network equations) using quantifier elimination methods. However, this is only feasible in the case of very small networks or systems with special structure. For more complex systems we need to compute approximate solutions, possibly using numerical computations.

In our situation we need compute discrete time trajectories for hybrid-DAE systems, both forward in time (for the control

problem) and backward in time (for the estimation problem). The distinguishing feature of hybrid systems vis-à-vis systems with only continuous dynamics is that not all of the dependent variables need be continuous in time. Ordinarily, there will be discontinuities at time of discrete state transitions. In the following paragraphs we describe a discrete time model for the forward case. The backward model is obtained in a similar way.

A. Problem Definition

Consider the semi-explicit Differential-Algebraic-Equation (DAE)

$$\begin{aligned} \dot{x} &= f(x,y,u) \\ 0 &= g(x,y,u) \end{aligned} \tag{1}$$

where time $t \in R^+$, $u \in R^m$ is an external input, the state is composed of $x \in R^n, y \in R^p$ and the functions $f : R^{n+p+m} \to R^m, g : R^{n+p+m} \to R^p$ define the evolution of the state. We will assume that the control $u(t)$ is piecewise continuous and the state $x(t)$ is continuous. Our goal is to show that under appropriate conditions the system (1) can be approximately described by a discrete time system

$$\begin{aligned} \hat{x}_{k+1} &= F_1(\hat{x}_k, \hat{y}_k, u_k) \\ \hat{y}_{k+1} &= F_2(\hat{x}_k, \hat{y}_k, u_k) \end{aligned} \tag{2}$$

Where $\hat{x}_k, \hat{y}_k$ are approximations to $x(t_k), y(t_k)$, respectively.

B. Differential Algebraic Equations – Hybrid Case

We will derive a discrete time representation where $t_k = kh, k = 0,1,2,\ldots$ and $h > 0$ is the time increment. It is assumed that $u(t) = u_k$, a constant, for $t \in [t_k, t_{k+1})$. A basic assumption is that mode changes can occur only at the time instants, t_k. In other words, for the interval $t \in [t_k, t_{k+1})$ we the system is described by

$$\begin{aligned} \dot{x} &= f_{i(t_k)}(x,y,u) \\ 0 &= g_{i(t_k)}(x,y,u) \end{aligned} \tag{3}$$

Where $i \in I_{\text{mode}}$, the mode index set. If we disallow resets during mode transitions, then it is reasonable to assume that $x(t)$ is continuous. On the other hand, $y(t)$ cannot be expected to be continuous across a mode transition. This implies that at each discrete time instant t_k it is necessary to compute $y_k^+ = y(t_k^+)$ from

$$0 = g_{i(t_k)}(x_k, y_k^+, u(t_k^+)) = g_{i(t_k)}(x_k, y_k^+, u_k)$$

We will drop the subscript $i(t_k)$ which is to be understood in the following expressions. Thus, y_k^+ is obtained from:

$$0 = g(x_k, y_k^+, u_k) \tag{4}$$

Now, we use a backward difference formula, in particular the trapezoidal formula, to obtain from (1):

$$\begin{aligned} x_{k+1} &= x_k + \tfrac{1}{2}h\left(f(x_k, y_k^+, u_k) + f(x_{k+1}, y_{k+1}, u_k)\right) \\ 0 &= g(x_{k+1}, y_{k+1}, u_k) \end{aligned} \tag{5}$$

With x_k, y_k^+, u_{k+1} known, it is necessary to solve (5) for x_{k+1}, y_{k+1}. To do this Taylor expand $f(x_{k+1}, y_{k+1}, u_k)$ and $g(x_{k+1}, y_{k+1}, u_k)$ about an initial estimate x_{k+1}^0, y_{k+1}^0 to obtain

$$\begin{aligned} x_{k+1} = x_k &+ \tfrac{h}{2}\left(f(x_k, y_k^+, u_k) + f(x_{k+1}^0, y_{k+1}^0, u_k)\right) \\ &+ \tfrac{h}{2}\frac{\partial f}{\partial x}(x_{k+1}^0, y_{k+1}^0, u_k)(x_{k+1} - x_{k+1}^0) \\ &+ \tfrac{h}{2}\frac{\partial f}{\partial y}(x_{k+1}^0, y_{k+1}^0, u_k)(y_{k+1} - y_{k+1}^0) + h.o.t \\ 0 = g(x_{k+1}^0, y_{k+1}^0, u_k) &+ \frac{\partial g}{\partial x}(x_{k+1}^0, y_{k+1}^0, u_k)(x_{k+1} - x_{k+1}^0) \\ &+ \frac{\partial g}{\partial y}(x_{k+1}^0, y_{k+1}^0, u_k)(y_{k+1} - y_{k+1}^0) + h.o.t \end{aligned}$$

By neglecting higher order terms, these equations can be approximately solved for the unknowns x_{k+1}, y_{k+1} using the linear equations:

$$\begin{bmatrix} I - \frac{h}{2}\frac{\partial f}{\partial x}(x_{k+1}^0, y_{k+1}^0, u_k) & -\frac{h}{2}\frac{\partial f}{\partial y}(x_{k+1}^0, y_{k+1}^0, u_k) \\ -\frac{\partial g}{\partial x}(x_{k+1}^0, y_{k+1}^0, u_k) & -\frac{\partial g}{\partial y}(x_{k+1}^0, y_{k+1}^0, u_k) \end{bmatrix} \begin{bmatrix} (x_{k+1} - x_{k+1}^0) \\ (y_{k+1} - y_{k+1}^0) \end{bmatrix} = \begin{bmatrix} (x_k - x_{k+1}^0) + \frac{h}{2}\left(f(x_k, y_k^+, u_k) + f(x_{k+1}^0, y_{k+1}^0, u_k)\right) \\ g(x_{k+1}^0, y_{k+1}^0, u_k) \end{bmatrix} \tag{6}$$

The approximation can be improved by replacing the initial values with the solution of (6), $x_{k+1}^0 \leftarrow x_{k+1}$, $y_{k+1}^0 \leftarrow y_{k+1}$ and resolving (6). Continuing recursively in this way is, of course, the Newton-Raphson method for finding solutions of (5).

If h is small and $x(t)$ continuous, it is common to take $x_{k+1}^0 = x_k, y_{k+1}^0 = y_k^+$, so that (6) becomes

$$\begin{bmatrix} I - \frac{h}{2}\frac{\partial f}{\partial x}(x_k, y_k^+, u_k) & -\frac{h}{2}\frac{\partial f}{\partial y}(x_k, y_k^+, u_k) \\ -\frac{\partial g}{\partial x}(x_k, y_k^+, u_k) & -\frac{\partial g}{\partial y}(x_k, y_k^+, u_k) \end{bmatrix} \begin{bmatrix} (x_{k+1} - x_k) \\ (y_{k+1} - y_k^+) \end{bmatrix} = \begin{bmatrix} h f(x_k, y_k^+, u_k) \\ g(x_k, y_k^+, u_k) \end{bmatrix} \tag{7}$$

Note that if a mode transition takes place at t_k into a mode requiring a state reset to say x^*, then we would take $x_{k+1}^0 = x^*$. In summary the discrete time model is given by (4) and (6) or (7). Notice that by making appropriate correspondences, the model (4) and (7) does take the form of (2).

Ordinarily, the model is integrated by solving (4) for y_k^+ using a Newton-Raphson method, with termination dependent on an error check, or simply a fixed number of iterations. The latter is typical for DAE solvers. Then the linear Equation (7)

is solved for x_{k+1}, y_{k+1}. If we choose to implement the computations in the most flexible manner, we should permit solution of (4) with any specified number of iterations, n_1 and then x_{k+1}, y_{k+1} should be obtained from (6) with any specified number of iterations, n_2.

Algorithm 1:
Given x_k, y_k, u_k and n_1, n_2 and $f, f_x, f_y; g, g_x, g_y$
Determine x_{k+1}, y_{k+1}

1. Compute y_k^+ from $g(x_k, y_k^+, u_k) = 0$ using n_1 iterations of Newton's method, starting with y_k
   a. Newton$\left(g(x_k, y, u_k), g_y(x_k, y, u_k), y, y_k^+, n_1\right)$
2. Compute x_{k+1}, y_{k+1} from

$$x_{k+1} - x_k - \tfrac{1}{2}h\left(f(x_k, y_k^+, u_k) + f(x_{k+1}, y_{k+1}, u_k)\right) = 0$$
$$g(x_{k+1}, y_{k+1}, u_k) = 0$$

   using n_2 iterations of Newton's method starting with x_k, y_k^+
   a. Define $z = (x, y)$,

$$F(z) = \left(x - x_k - \tfrac{1}{2}h\left(f(x_k, y_k^+, u_k) + f(x, y, u_k)\right), g(x, y, u_k)\right)$$

and

$$F_z = \begin{bmatrix} I - \frac{h}{2} f_x(x, y, u_k) & -\frac{h}{2} f_y(x, y, u_k) \\ g_x(x, y, u_k) & g_y(x, y, u_k) \end{bmatrix}$$

   b. Newton$\left(F(x, y, u_k), F_z(x, y, u_k), (x, y), (x_k, y_k), n_2\right)$

The idea is that for any given power system in the form of (1) we will automatically create code for implementing (4) and (6), with n_1, n_2 as parameters. The computational routines will be targeted for numerical computation in Mathematica and Simulink. We developed Mathematica code to implement the following functions.

Function Newton:
Newton$\left(F(x), F_x(x), x, x_0, n\right)$
Given: function $F(x)$, Jacobian $F_x(x)$, initial estimate x_0 and number of iterations n

1. set $x_I = x_0$, $k = 1$
2. While $k \le n$
   a. solve for x, $F(x_I) + F_x(x_I)(x - x_I) = 0$ (use *Mathematica* function Solve)
   b. set $x_I = x$
   c. set $k = k + 1$

Function DAEDiscrete:
DAEDiscrete$\left(f, g, x, y, x_0, y_0, u, n_1, n_2, h\right)$

1. Given:
   a. Functions f, g
   b. Argument lists x, y, u, x_0, y_0
   c. Iteration integers n_1, n_2
   d. Time increment h
2. Implement Algorithm 1 to compute $F(z,u)$ such that

$$z_{i+1} = F(z_k, u_k), \quad z_k = (\hat{x}_k, \hat{y}_k) \tag{8}$$

C. Utilization

If x_0, y_0, u are symbols, then (8) gives us the discrete time approximation suggested in (2). This is the primary intent of this model. Ordinarily, we take h small and expect that a relatively small number of iterations, i.e., n_1, n_2 are small. Once F is computed, it can be simplified using standard Mathematica functions. In view of the expected complexity of the expressions, it may also be desirable to truncate them, retaining only low orders of the small parameter h.

V. EXAMPLE

Consider the system shown in Figure 1. In the event of a failure of generator 2 the system has the structure shown in Figure 3. The system may be configured in three different ways: 1) as shown, 2) with the vital load (dashed box) fed from bus 6 instead of 3, or with the vital load disconnected from the network and fed from the battery.

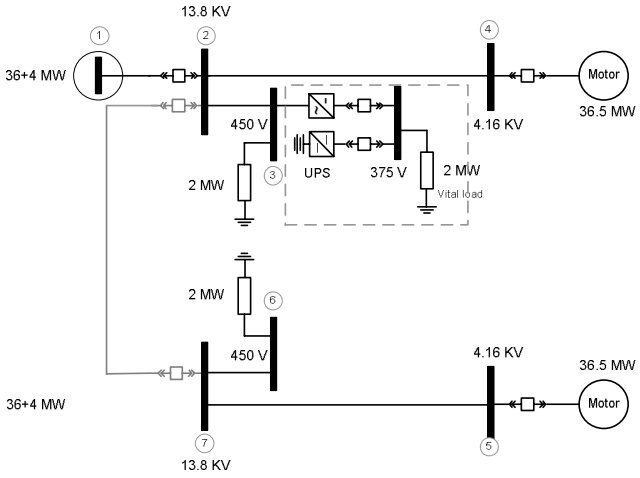

Figure 3. System following loss of Generator 2.

We need only model the system shown as the other cases follow easily.

Notice that all of the loads other than the vital load are constant admittance loads. This includes the motors, each of which can be considered a constant admittance load with a slowly varying parameter (slip). The vital load is fed from a bus (bus 3 in Figure 3) through a converter which we consider to power factor corrected. Thus, the load appears to the AC side as constant power with unity power factor. Consequently, the system can be reduced to the three bus network as shown in Figure 4. In fact the generator terminal bus is retained only because it is voltage controlled.

The reduced bus admittance matrix is

$$Y_R = \begin{bmatrix} -ib_{11} & ib_{21} & 0 \\ ib_{21} & -g_{22} - ib_{22} & g_{23} + ib_{23} \\ 0 & g_{23} + ib_{23} & -g_{33} - ib_{33} \end{bmatrix} \tag{9}$$

where

$$\begin{gathered} b_{11} = b_{21} = b_{23} = 1.5, \quad b_{33} = 1.5556 \\ g_{23} = 0, g_{33} = 0.0556 \end{gathered} \tag{10}$$

and

$$\begin{aligned} b_{22} &= 4.555 - \frac{3.375 + 2.25d_1}{c_1^2 + (1.5 + d_1)^2} \\ &\quad - \frac{0.1081 - 1.385c_2 + 2.0856d_2}{0.00533 + c_2(0.09586 + c_2) + d_2(0.1100 + d_2)} \\ g_{22} &= -\frac{2.25d_1}{c_1^2 + (1.5 + d_1)^2} \\ &\quad - \frac{0.1076 + 2.0856c_2 + 0.1385d_2}{0.00533 + c_2(0.09586 + c_2) + d_2(0.1100 + d_2)} \end{aligned} \tag{11}$$

Where c_1, d_1 and c_2, d_2 are functions of the motor slips s_4 and s_5, respectively.

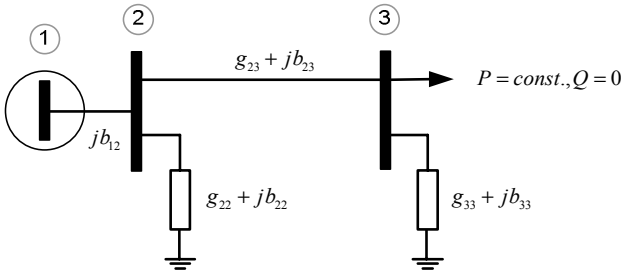

Figure 4. Reduced network obtained by eliminating internal buses.

The reduced network equations are comprised of the real and reactive equations for buses 2 and 3:

$$\begin{aligned} 0 &= g_{22}V_2^2 - g_{23}V_2V_3\cos\theta_{23} - b_{21}E_1V_2\sin\theta_2 \\ &\quad - b_{23}V_2V_3\sin\theta_{23} \\ 0 &= -P_v + g_{33}V_3^2 - g_{23}V_2V_3\cos\theta_{23} + b_{23}V_2V_3\sin\theta_{23} \\ 0 &= -b_{22}V_2^2 + b_{21}E_1V_2\cos\theta_2 + b_{23}V_2V_3\cos\theta_{23} \\ &\quad - g_{23}V_2V_3\sin\theta_{23} \\ 0 &= -b_{33}V_3^2 + b_{23}V_2V_3\cos\theta_{23} + g_{23}V_2V_3\sin\theta_{23} \end{aligned} \tag{12}$$

The slip equations are

$$\begin{aligned} \dot{s}_4 &= \frac{1}{4}\left(0.7 - \frac{0.25(1 - s_4)s_4V_4^2}{0.0625 + 0.15625s_4^2}\right) \\ \dot{s}_5 &= \frac{1}{4}\left(0.7 - \frac{0.25(1 - s_5)s_5V_5^2}{0.0625 + 0.15625s_5^2}\right) \end{aligned} \tag{13}$$

with $V_4 = f_4(c_1, c_2, d_1, d_2)V_2$ and $V_5 = f_5(c_1, c_2, d_1, d_2)V_2$. Equations (13) are the differential equations and equations (12) the algebraic equations. This semi-explicit DAE has state s_1, s_2 and dependant variables $V_2, V_3, \theta_2, \theta_{23}$. The function DAEDiscrete can now be applied.

VI. Conclusions

Modern power system management problems require the ability to address the interaction of complex nonlinear dynamics and discrete events. We have described an approach to building models of power systems that involve classical semi-explicit power system dynamics along with discrete events. These models are intended to be used for building simulations in SIMULINK with Stateflow, where the latter is used to represent the finite state machine that characterizes the discrete event dynamics. These same models can also be used to design optimal power system management strategies along the lines described in [1, 2].

The basic idea is to convert the semi-explicit DAE that characterizes the power system continuous dynamics to a system of discrete time difference equations that propagates the state and other dependent variables forward in time in a way that allows for the occurrence discrete transitions at any discrete time step.

VII. References

[1] H. Kwatny, E. Mensah, D. Niebur, and C. Teolis, "Optimal Shipboard Power System Management via Dynamic Mixed Integer Programming," in *IEEE Electric Ship Technologies Symposium*, Philadelphia, 2005.

[2] H. G. Kwatny, E. mensah, D. Niebur, and C. Teolis, "Optimal Power System Management via Mixed Integer Dynamic Programming," in *2006 IFAC Symposium on Power plants and Systems*, Kananaskis, Canada, 2006.

[3] SYNTEK, "CAPS DD IPS Electrical Distribution One-Line Diagram," 2003.

[4] SYNTEK, "DD(X) Notional Baseline Modeling and Simulation Development," SYNTEK Technologies, Arlington August 1, 2003 2003.

[5] K. E. Brenan, S. L. Cambell, and L. R. Petzold, *Numerical Solution of Initial–Value Problems in Differential–Algebraic Equations*. New York: Elsevier Science Publishing Co., 1989.

This work was supported in part by the Office of Naval Research under contract # N00014-06-C-0041.