Mechatronics 22 (2012) 83–94

Contents lists available at SciVerse ScienceDirect

Mechatronics

journal homepage: www.elsevier.com/locate/mechatronics

Design evolution of mechatronic systems through modeling, on-line monitoring, and evolutionary optimization

L.B. Gamage, C.W. de Silva *, R. Campos

Industrial Automation Laboratory, Department of Mechanical Engineering, The University of British Columbia, Vancouver, Canada

ARTICLE INFO

Article history:
Received 19 April 2011
Accepted 30 November 2011
Available online 13 January 2012

Keywords:
Design evolution
Linear Graphs
Electro-mechanical modeling
Genetic programming

ABSTRACT

This paper presents a system framework to automate the design evolution of multi-domain engineering systems. The developed system integrates machine health monitoring with an expert system to monitor the performance of an existing mechatronic system and to take a decision as to whether and which part of the system a design improvement is necessary. A system model represented by Linear Graphs is evolved using genetic programming to obtain a design that optimally satisfies a specified level of performance. The developed approach allows exploration of the design space to find the optimum design solution in an automated manner. In order to control the arbitrary exploration of the design space, domain knowledge, expertise, and input from the machine health monitoring system are used. The design evolution algorithm is implemented using GPLAB, a MATLAB tool, and integrated with Simscape for modeling and simulation. The developed system is applied to redesign the electro-mechanical conveyor system of an industrial fish processing machine.

1. Introduction

Design of multi-domain engineering systems can be rather complex and it requires an integrated and concurrent approach to obtain optimum results. The traditional approach is sequential where subsystems are designed separately in their own domains neglecting their dynamic interactions with each other. Also, the current practices in industry focus only on the final phases of the design – the detailed design or parametric design to improve the performance and comply with design specifications. Even though some progress has been made in the area of integrated and multi-domain optimal design further work with direct practical relevance is needed [1–6].

In the context of design optimization a major challenge has been the lack of schemes that can accommodate high complexity and non-analytic and multiple design objectives. The use of genetic programming [7] has alleviated this problem to a great extent, and is able to accomplish evolutionary optimization so as to satisfy a set of specified design objectives. Koza et al. [6] has applied genetic programming for the automated design of electrical circuits. Inspired by Koza's work, Seo et al. [5] introduced the concept of combining bond graph (BG) modeling with Genetic Programming (GP) as a means of exploring the design space of a mechatronic system. Behbahani and de Silva [2–4] have used genetic programming and bond graphs for the identification of the topology and the parameter values of a mechatronic system. Hu et al. [8] have applied BG and GP for automated synthesis of mechatronic systems.

Evolution of a design model (e.g., BG) of a mechatronic system through genetic programming allows exploration of the design space for an optimum solution in an automated manner. However, there are issues that need to be addressed before this approach can be used for design improvement and implementation of practical and complex mechatronic systems. For example, arbitrary evolution of a design model may result in complex designs that are not feasible in practical implementations. Domain knowledge and expertise and information gathered through a machine health monitoring system are incorporated in the present work to address these issues. In particular, this paper develops a system framework for automated evolution of an optimal design for an existing multi-domain (mechatronic) engineering system. The developed evolutionary design framework integrates a Machine Health Monitoring System (MHMS), a model generation system, a Design Expert System (DES) and an Evolutionary Design Optimization System (EDOS) as shown in Fig. 1, and will assist engineers in the design/redesign of mechatronic systems. A machine health monitoring system maintains an engineering system in an operable condition by predicting, detecting, diagnosing, and correcting system faults and malfunctions [9,10]. The MHMS acquires data from a number of different sensors and monitors the current state of the engineering system. In the system framework presented in this paper the MHMS provides vital information to the DES particularly regarding potential regions of design weakness. In the present context, the model generation system provides an appropriate representation

* Corresponding author.
E-mail address: desilva@mech.ubc.ca (C.W. de Silva).

0957-4158/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.mechatronics.2011.11.012

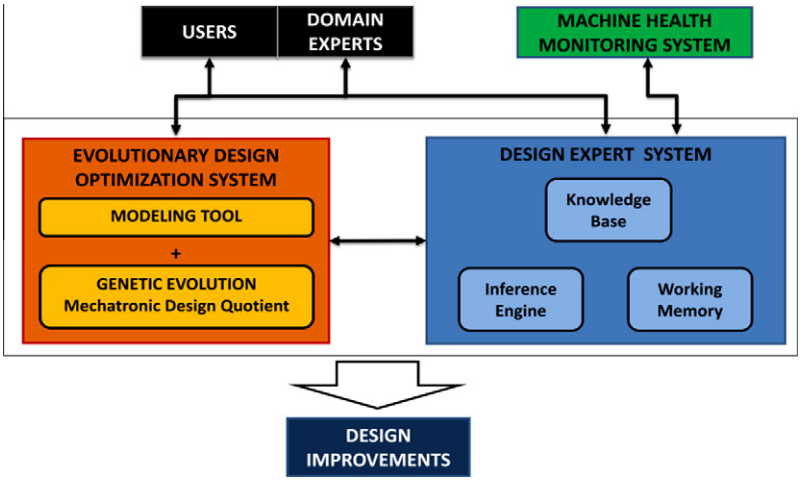

Fig. 1. The system framework of design evolution.

of the engineering system, which is analyzed to make design changes such that the resulting physical system performs according to the required specifications. Since mechatronic systems are considered, it is important that the model generation system is capable of modeling a multi-domain engineering system in a unified manner. Several modeling tools such as bond graphs [2,4,5,12] and Linear Graphs may be used for this purpose. LGs as opposed to BGs provide a realistic representation resembling the physical structure of a mechatronic system. For example, parallel mechanical elements correspond to series electrical elements in BG and whereas parallel electrical elements in LG. Also, BGs require additional computations to check for causality conflicts whereas LGs by nature do not have causality conflicts [3,11,12]. The Linear Graph approach has been adopted in the present work mainly due the above mentioned advantages and the availability of a systematic methodology to obtain a unique state space model [3,12].

The Design Expert System provides expert input to the evolutionary framework. The DES in addition to receiving and storing information from the MHMS, also captures knowledge from human experts for the necessary decision making [7]. The Evolutionary Design Optimization System automatically evolves a multi-domain mechatronic design (represented by a Linear Graph) using genetic programming. The design configurations that are generated in this manner satisfy the specified performance requirements. The performance requirements may be represented by a mechatronic design quotient or MDQ [13,14]. The evolution process changes both the structure and the parameter values of the system to produce an optimal design.

2. Evolutionary Design Optimization System

The development of the present approach starts with a representation of the mechatronic system, which is under consideration. A Linear Graph (LG) model will be employed for this purpose for its advantages in representing mechatronic systems. In each stage of design evolution, the LG model will be used for analysis and simulation of the mechatronic system.

2.1. Linear Graphs

Linear Graphs use interconnected line segments called branches to represent ideal lumped-parameter elements [3,10,11]. Each branch has an ordered pair of variables associated with it: a *through variable* (f) and an *across variable* (v). The product of the two variables is the power variable. Every branch is oriented and

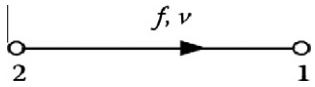

Fig. 2. Linear Graph branch of an element.

Table 1
Through and across variables associated with several domains.

Domain	Through variable	Across variable
Electrical	Current	Voltage
Mechanical	Force/torque	Velocity/angular velocity
Hydraulic/pneumatic	Flow rate	Pressure
Thermal	Heat transfer	Temperature

the branches are connected at nodes. One end of a branch is the *point of action* and the other end is the *point of reference*. An arrowhead pointing toward the *point of reference* is used to describe the positive direction of flow (see Fig. 2).

The value of the *through variable* (f) does not change through an element at a given time. However, the absolute value of the *across variable* (v) does change "across" the element at a given time, and the value at the point of action relative to that at the point of reference is defined as the value of the across variable v. Table 1 presents the typical through and across variables associated with some common domains.

In an LG there are four types of elements. Single-port passive elements are C (across-type or A-type energy storage element), I (through-type or T-type energy storage element) and R (dissipation-type or D-type element) which in the electrical domain represent capacitors, inductors and resistors and in the mechanical domain represent inertia (mass), spring and damper, respectively. Single-port active elements are source elements, which may be classified as *T-Type Source* (Current Source, Force Source) and *A-Type Source* (Voltage Source, Velocity Source).

There are two Two-port elements representing transformers and gyrators in which power is conserved. The *transformer* element is an ideal element where the across and through variables associated with one branch are transformed into another pair of values without dissipating or storing energy. Fig. 3 shows the Linear Graph representation of a *transformer* where v_i and f_i are the across and through variables of the input port, and v_o and f_o are the across and through variables of the output port. For the transformer r is the transformation ratio between the two across variables of the

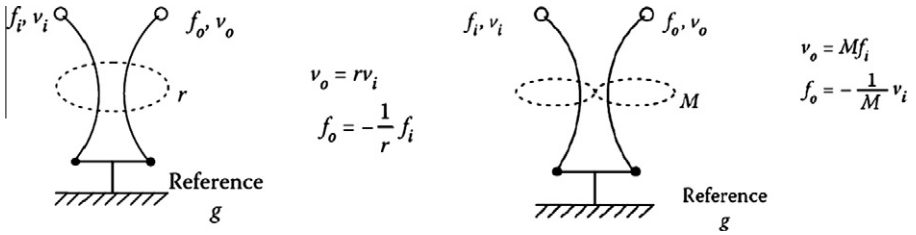

Fig. 3. Linear Graph representation of a transformer and a gyrator.

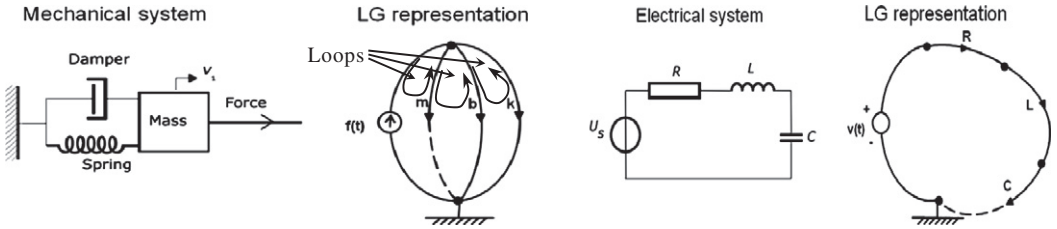

Fig. 4. LG Representation of a mechanical oscillator and an electrical circuit.

branches (and equivalently, for the corresponding two through variables). For the gyrator, M is the parameter that relates the through variable of one branch to the across variable of the other branch.

Fig. 4 shows the LG representations of a simple mechanical system in parallel configuration and an electrical system in series configuration.

In order to obtain an analytical model from a Linear Graph three types of equations have to be established.

- Constitutive (physical) equations.
- Compatibility (or loop) equations.
- Continuity (or node) equations.

The constitutive equations correspond to the "physical" equations of all the branches (elements) except for sources (inputs). The compatibility equations correspond to the independent loops in the Linear Graph. A loop is a closed path formed by at least two branches as shown in Fig. 4. The compatibility equations are obtained by summing to zero all the across variables along the branches that form each "independent" loop. Compatibility assures system integrity in that at a given instant and at a given point in the system the across variable value is unique (i.e., there are no discontinuities). A sign convention dictates that the direction of the branch arrow is the direction of drop of the across variable, which is considered positive, except for a T-type source (e.g., a current source) where the arrow is in the positive direction of the through variable that is coming out of the source, and hence it is also in the direction of "increase" of the across variable. Continuity equations are obtained by summing to zero all the through variables at each "independent" node of the LG (the direction "into" the node is taken positive). These equations represent such conditions as force balance at a point in a mechanical system, current balance at point in an electrical system, and flow continuity at a point in a hydraulic system. For example, for the LG of the electrical circuit shown in Fig. 4, the following equations can be written:

Constitutive	Node	Loop
$V_R = Ri_R$	$Node1 : i - i_R = 0$	$V(t) - V_R - V_L - V_C = 0$
$V_L = L\frac{di_L}{dt}$	$Node2 : i_R - i_L = 0$	
$i_L = C\frac{dV_c}{dt}$	$Node3 : i_L - i_C = 0$	

Once the constitutive, compatibility and continuity equations are written, the systematic methodology to obtain the state space model comprises the following steps:

1. Select the state variables for the independent elements: across variables for the independent A-type elements and through variables for the independent T-type elements.
2. Write the state space shell. The state space shell is formed by the constitutive equations of the independent energy storage elements.
3. Write the remaining constitutive equations for the other elements.
4. Write the continuity equations for the primary (i.e., independent) nodes.
5. Write the compatibility equations for the primary (i.e., independent) loops.
6. Finally, eliminate all the other variables in the state space shell which are not state variables or input variables, using the node, loop and remaining constitutive equations.

2.2. Design evolution

Genetic Programming (GP) is a branch of evolutionary computing and is an extension of Genetic Algorithms (GAs) [7,15,16]. In GA a solution is represented by a chromosome, typically a sequence of ones and zeros of fixed length whereas in GP a solution is represented by a tree like structure of variable length and topology allowing a solution to grow. The growth of the solution is achieved by a set of construction functions and terminals which are added to modifiable sites so as to optimize a fitness function (MDQ in the present application). A population is made up of a number of trees (chromosomes) and evolution is carried out by the genetic operations of selection, cross-over and mutation [2,5,7]. A tree is grown from an *embryo model* which reflects the fundamental specifications of the solution and is common to all solutions.

In the present work, a methodology is developed to evolve Linear Graphs using genetic programming. First a set of construction functions and terminals are established to evolve an LG using GP. A construction function adds a component to a modifiable node of the evolving model and assigns a value to it whereas a terminal ends further evolution of an LG thereby controlling arbitrary

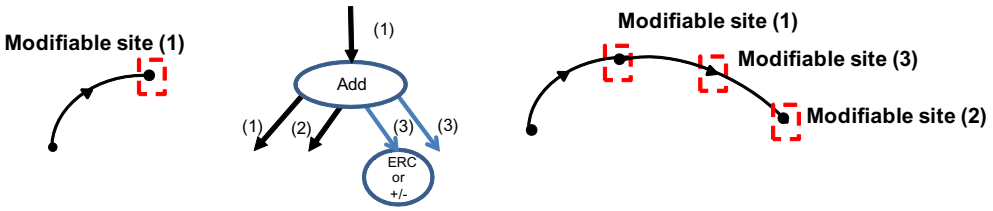

Fig. 5. Illustration of adding an element to a modifiable node.

Table 2
LG-GP functions and terminals.

Name	Arity	Description
Add_A	1	Adds an A-type element (Mass, Moment of Inertia, Capacitor, Thermal Capacitor, or Fluid Capacitor)
Add_T	1	Adds a T-type element (Linear Spring, Torsional Spring, Inductor, or Fluid Inertor)
Add_D	1	Adds a D-type element (Viscous Damper, Resistor, Thermal Resistor, or Fluid Resistor)
Add_t	1	Adds a transformer type element (two-port)
Add_g	1	Adds a Gyrator type element (two-port)
Add_L	2	Adds a line between two modifiable nodes
Replace_A	2	Replaces the current element with an A-type element
Replace_T	2	Replaces the current element with a T-type element
Replace_D	2	Replaces the current element with a D-type element
Replace_t	2	Replaces the current two-port element with a transformer type element
Replace_g	2	Replaces the current two-port element with a Gyrator type element
Desprog_2	0	Executes two functions sequentially
Desprog_3	0	Executes three functions sequentially
Close_node	1	Terminates a modifiable node
Gnd	1	Connects a modifiable node to ground reference and terminates it
ERC	0	Ephemeral random constant
+, −, ∗	2	Adds, subtracts, or multiplies two ERCs
End	0	Terminates a modifiable branch

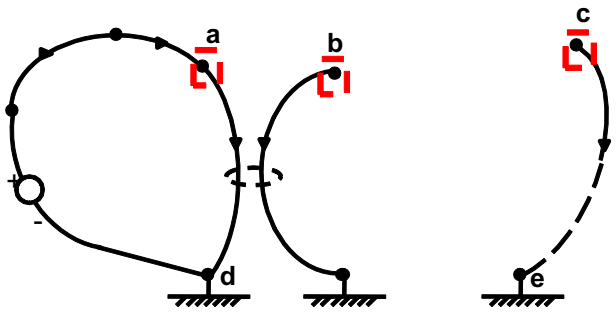

Fig. 6. Embryo example of a mechatronic LG model.

evolution. Fig. 5 illustrates how the Add function inserts an element into a modifiable site (node) of an LG and creates two new modifiable sites. The add function maintains the modifiable site 1 for further possible modification at the node. It creates a new node at the modifiable site 2 and will add a single-port A-, T-, or D-type element at modifiable site 3. The modifiable site 3 allows the branch to change its parameter value.

Table 2 gives the functions and the terminals used in the present work. In particular, note that the *Add_A* function has a special meaning in the mechanical domain. When it is a mass or a moment of inertia element, the other terminal of the element has to be connected to a reference ground.

The evolution of a mechatronic system by using the functions and terminals given in Table 2 may be illustrated as follows. Consider the embryo LG model shown in Fig. 6. On-line information from the MHMS may indicate an underperformance with respect to a set of specifications or the occurrence of a malfunction or a fault possibly due to a design weakness. The Design Expert System, after ruling out a component malfunction or failure, isolates the region of design weakness, with the assistance of the MHMS [10,17]. Evolution is allowed only from the modifiable sites marked (a), (b), and (c) which correspond to the regions identified by the DES. Restricting the modifiable sites to the identified areas of the LG with the help of the available knowledge will guide the design exploration thereby avoiding arbitrary evolution.

Fig. 7 illustrates how the embryo shown in Fig. 6 is evolved to produce the LG model shown in Fig. 8. The evolution process begins by adding a "T-type" element to node (a) with an *Add_T* function. It will generate a branch, determine its parameter value and will connect the branch to the ground reference by applying a *Gnd* terminal. The evolution process of this added branch terminates with an *End* operator and a *Close_node* terminal so that no further modification occurs at that node. Next, an *Add_A* function is inserted to node (b), the parameter values are calculated and the branch modifiable site is terminated with an *End* operator. In this case, the node (b) continues to be modifiable and from the same node an *Add_D* function as well as an *Add_t* function is executed one after the other. Once the *Add_t* function has been executed, a *Close_node* terminal is applied disallowing any possibility of further modification.

Now, from the modifiable site generated by the *Add_t* function at node (4) of Fig. 7, another *Add_T* function and an *Add_D* function are executed. In both cases, the parameter values are calculated using *ERC*s, + and – terminals, and the modifiable site at each branch is terminated with an *End* operator. Notice that the *Add_T* function and the *Add_D* function in this section of the GP tree use the *Desprog_2* and *Desprog_3* functions together with *Add_L* function to perform several operations sequentially. In particular, they connect branches from node (3) to node (c) and close the nodes (3) and (c).

The algorithm of the present framework of design evolution is shown in Fig. 9. The dark shaded blocks identify steps where the MHMS and the DES exchange information so that the evolutionary

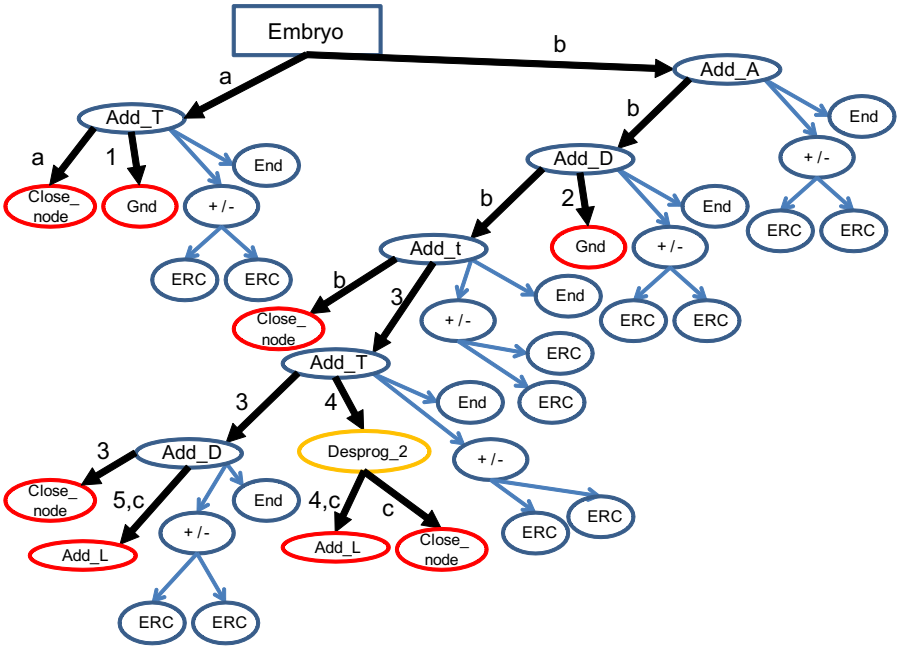

Fig. 7. Illustration of an LG growth from an Embryo model.

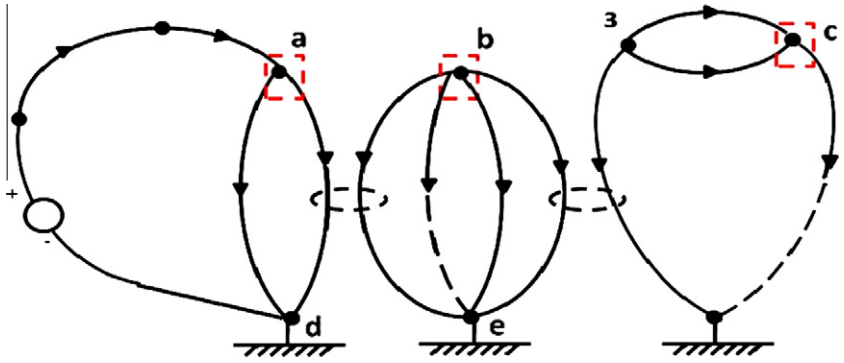

Fig. 8. LG model evolved by the GP tree.

process (EDOS) can take place. Operations executed by the EDOS such as LG model creation, embryo model initialization with modifiable sites, evolution of alternatives, and simulation of feasible design candidates are indicated in light shaded blocks. Clear blocks correspond to the activities performed in the experimental phase of the research.

2.3. Fitness function

The purpose of the fitness function or fitness measure is to specify what the user wishes to achieve in the optimization. It is a measure of the closeness of the evolved solution to the desired solution. Hence, for every individual design solution, the fitness calculation must reflect how well its output resembles the desired output. In general, there may be several parameters or specifications that characterize the desired solution. The value of the fitness function must be computed for each evolved individual while taking into account the closeness of each parameter of the evolved individual to the corresponding desired parameter. In the present work, the fitness measure uses a raw fitness $r(i,t)$ of the individual i at any generational step t, which is defined as the absolute sum over N_e parameters of the error (difference) between the desired system output $C(j)$ for parameter j and the evaluated response $S(i,j)$ of individual i for parameter j:

$$r(i,t) = \sum_{j=1}^{N_e} |S(i,j) - C(j)| \quad (1)$$

In the *raw fitness* calculation, it is clear that a smaller fitness measure is better and a fitness measure of 0 will indicate a perfect match with the target.

3. Implementation in the experimental system – the intelligent iron butcher

The test bed that is used to validate the approach developed in the present work is the Intelligent Iron Butcher (IIB) shown in Fig. 10, which is an automated fish cutting machine developed in the Industrial Automation Laboratory (IAL) of the University of British Columbia. The purpose of the machine is to accurately remove the heads of Pacific salmon at high speed. The conventional machines used in the industry waste about 10–15% of useful meat, each unit percentage of wastage costing about $5 million annually in the province of British Columbia, Canada. In the IIB, a positioning

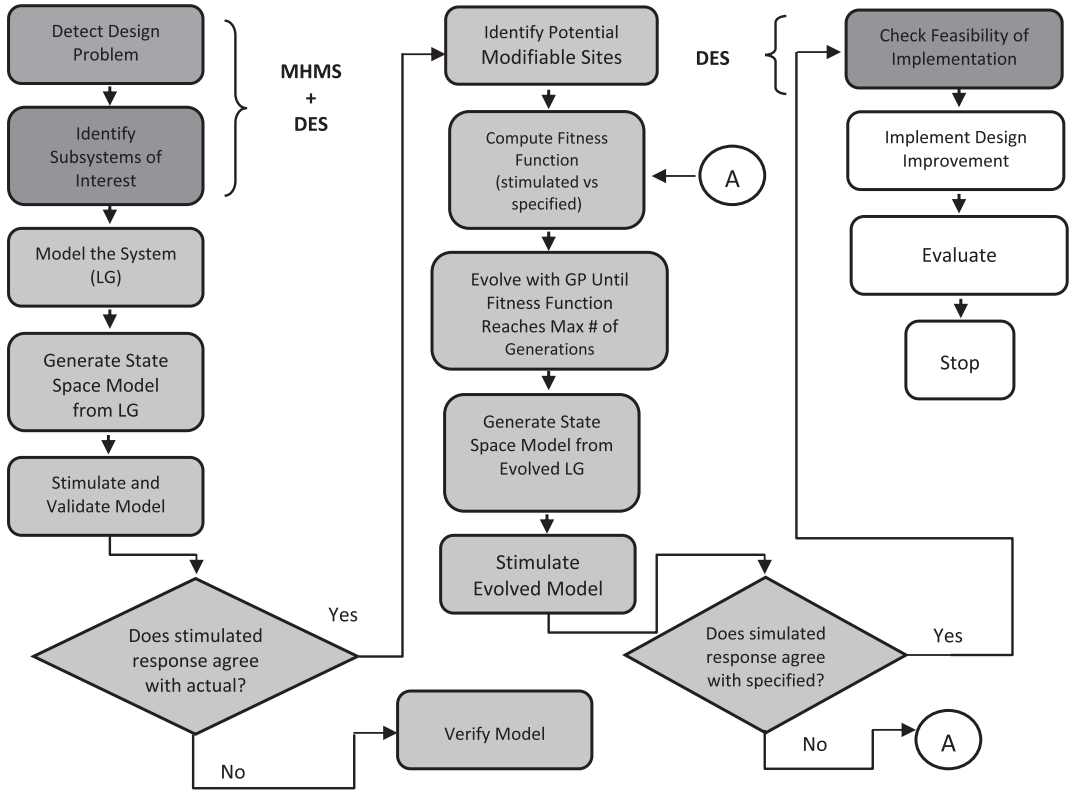

Fig. 9. Algorithm of the design evolution framework.

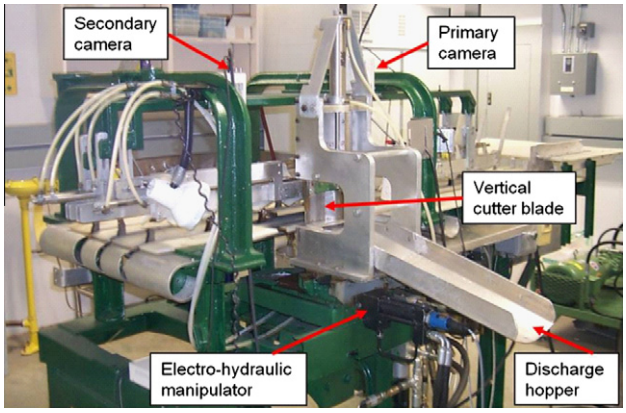

Fig. 10. Intelligent Iron Butcher (IIB).

and control system using hydraulic actuators, servo valves, and a computer vision system are employed to realize accuracies of the order of 1%. In the machine, the actual gill position of the fish is sensed and computed using a vision system. The cutter assembly, which is mounted on a two-axis positioning table, is actuated by two electro-hydraulic manipulators controlled by two servo-valves. The cutter is moved to the gill position, according to the computed value, so that the fish head can be removed with minimal wastage. This entire operation including the computation of the gill position by the vision system has to be carried out within 500 ms, satisfying the required feed rate of two fish per second. Another computer vision system captures images of fish immediately after the head removal operation to determine the quality and the accuracy of the cut. An electromechanical conveying system, driven by an AC induction motor, moves the fish forward in an intermittent manner so that the fish is kept stationery when the image is captured and also when the head is removed. A pneumatically operated cutter blade, which is formed in the shape of an average gill, is pushed down to remove the head along the collar bone.

The methodology developed in this paper is applied to improve the design of the electromechanical conveying system of the IIB. The conveying system is driven by an induction motor coupled with a Variable Diameter Pulley (VDP) drive as shown in Fig. 11. The VDP compensates for speed variations of the motor/load, because the distance between the pulleys and the length of the belt do not change. What changes is the effective diameter of the pulleys which in turn adjusts the overall transmission ratio. The VDP is connected to a gearbox which decreases the output velocity and increases the output torque. The resulting rotary motion is converted into a push pull stroke through a mechanical linkage. The mechanical linkage is connected to a sliding mechanism as shown in Fig. 11 which moves the fish only in one direction as a set of fingers fold or remain open depending on the cycle of the motion.

3.1. Modeling and simulation of the conveying system

A lumped-parameter Linear Graph model for the electromechanical conveying system of the IIB has been developed, as shown in Fig. 12.

The next step is to derive the state space model for this dynamic system. According to the systematic methodology as described before and the LG model previously generated for the IIB conveying system, the following are established.

Number of branches $b = 14$; Number of nodes $n = 7$; Number of primary loops $l = 8$; Number of sources, $s = 1$.

As each source branch has one unknown variable and all the other passive branches have two unknown variables each, the following values are noted:

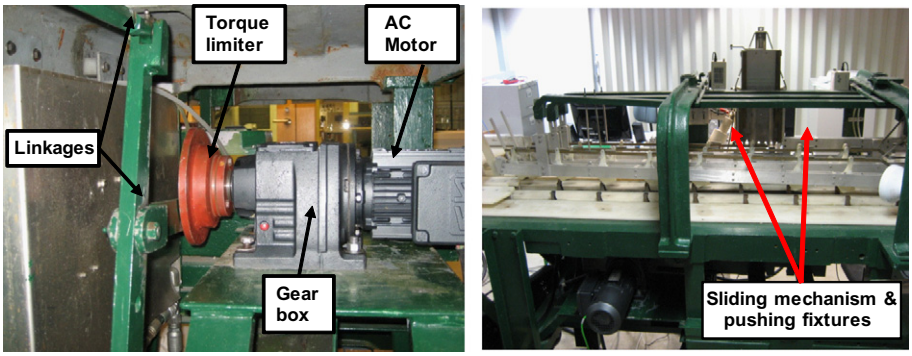

Fig. 11. The IIB conveying system with sliding mechanism.

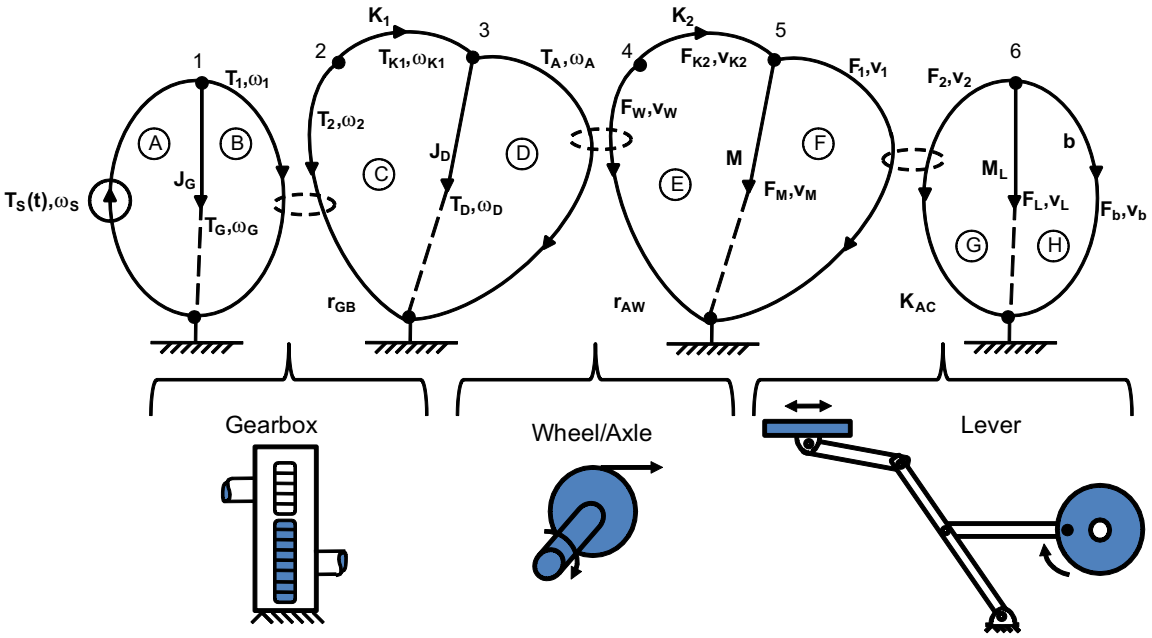

Fig. 12. LG of the electromechanical conveying system.

Number of unknown variables: $2b-s = 28-1 = 27$.
Number of constitutive equations: $b-s = 14-1 = 13$.
Number of independent node equations: $n-1 = 7-1 = 6$.
Number of loop equations: $b-n+1 = 14-7+1 = 8$.

Now as a verification, for the system to be solvable, the number of unknowns should be equal to the number of equations; thus: $2b-s = b-s+n-1+l$ or $27 = 13+6+8$.

The Linear Graph model corresponds to a sixth order system because it has six independent energy storage elements. The selected state variables are:

$$x = [x_1, x_2, x_3, x_4, x_5, x_6]^T = [\omega_G, \omega_D, T_{K1}, v_m, F_{K2}, v_L]^T$$

Hence the state space shell is formed by the following equations:

$$\dot{\omega}_G = \frac{1}{J_G}T_G, \quad \dot{\omega}_D = \frac{1}{J_D}T_D, \quad \dot{T}_{K1} = K_1\omega_{K1}, \quad \dot{v}_m = \frac{1}{M}F_m, \quad \dot{F}_{K2} = K_2 \cdot v_{K2}, \quad \dot{v}_L = \frac{1}{M_L}F_L$$

The remaining constitutive equations are:

$$F_b = b \cdot v_b, \quad v_1 = K_{AC} \cdot v_2, \quad F_2 = K_{AC} \cdot F_1, \quad v_W = r_{AW} \cdot \omega_A$$

$$T_A = r_{AW} \cdot F_W, \quad \omega_1 = r_{GB} \cdot \omega_2, \quad T_2 = r_{GB} \cdot T_1$$

The node equations are:

$$\text{Node 1} : T_s(t) - T_G - T_1 = 0, \quad \text{Node 2} : -T_2 - T_{K1} = 0, \quad \text{Node 3} : T_{K1} - T_D - T_A = 0$$

$$\text{Node 4} : -F_w - F_{K2} = 0, \quad \text{Node 5} : F_{K2} - F_M - F_1 = 0, \quad \text{Node 6} : -F_2 - F_L - F_b = 0$$

and the loop equations are:

$$\text{Loop A} : \omega_S - \omega_G = 0, \quad \text{Loop B} : \omega_G - \omega_1 = 0, \quad \text{Loop C} : \omega_2 - \omega_D - \omega_{K1} = 0, \quad \text{Loop D} : \omega_D - \omega_A = 0, \quad \text{Loop E} : -v_m - v_{K2} = 0, \quad \text{Loop F} : v_m - v_1 = 0,$$

$$\text{Loop G} : v_2 - v_L = 0, \quad \text{Loop H} : v_L - v_b = 0$$

where for gearbox, ω_1 = input shaft angular velocity; ω_2 = output shaft angular velocity; T_1 = torque on the input shaft; T_2 = torque on the output shaft; r_{GB} = Gear ratio.

For wheel/axle, v_w = linear velocity of the wheel periphery; ω_A = angular velocity of the axle; T_A = torque on the axle; F_W = force on the wheel periphery; r_{AW} = wheel radius.

For two-node first-class lever with fulcrum at node B, v_1 = lever joint input velocity; v_2 = lever joint output velocity; F_1 = lever joint input force; F_2 = lever joint output force

$$K_{AC} = \frac{l_{BC}}{l_{BC} + l_{AC}} \tag{2}$$

where l_{AC} and l_{BC} are the arm lengths. The driving motor is represented by an ideal torque source T_S with angular velocity ω_S. The following parameters are also defined: J_G = equivalent moment of inertia of the gear; J_D = equivalent moment of inertia of the wheel (disk); M = equivalent mass of the linkage; M_L = equivalent mass of the sliding copper bars (Load); K_1 = stiffness of the "flexible" shaft that couples the gearbox with the disk; K_2 = stiffness of the "flexible" shaft coupling the disk with the linkage; b = equivalent viscous damping constant; v_b = linear velocity of damping element; T_G = torque at the gear; T_{k1} = torque at the "flexible" shaft coupling the gearbox with the disk; T_D = torque at the wheel (disk); ω_G = angular velocity of the gear; ω_D = angular velocity of the wheel (disk);F_{K2} = force at the "flexible" shaft coupling the disk with the linkage; F_M = equivalent dynamic force of the linkage; F_L = equivalent dynamic force of the sliding copper bars (Load); F_b = force at the equivalent viscous damper; w_{k1} = angular velocity of the "flexible" shaft coupling the gearbox with the disk; v_m = linear velocity of the linkage; v_{k2} = velocity of the "flexible" shaft coupling the disk with the linkage; v_L = linear velocity of the sliding copper bars.

By eliminating the auxiliary variables in the state-space shell, the following state equations are obtained:

$$\dot{\omega}_G = \frac{1}{J_G}\left[T_S(t) + \frac{T_{K1}}{r_{GB}}\right], \quad \dot{\omega}_D = \frac{1}{J_D}[T_{K1} + r_{AW} \cdot F_{K2}], \quad \dot{T}_{K1} = K_1\left[\frac{\omega_G}{r_{GB}} - \omega_D\right]$$

$$\dot{v}_m = \frac{1}{M}\left[F_{K2} + \frac{b \cdot v_m}{K_{AC}^2} - \frac{b \cdot v_L}{K_{AC}}\right], \quad \dot{F}_{K2} = K_2[r_{AW} \cdot \omega_D - v_m], \quad \dot{v}_L = \frac{1}{M_L}\left[\frac{b \cdot v_m}{K_{AC}} - b \cdot v_L\right]$$

These represent the state-space model:

$$\begin{aligned} \dot{\boldsymbol{x}} &= \boldsymbol{Ax} + \boldsymbol{Bu} \\ \boldsymbol{y} &= \boldsymbol{Cx} + \boldsymbol{Du} \end{aligned} \tag{3}$$

The system output is $\boldsymbol{y} = [v_L]$

The corresponding system matrix is

$$A = \begin{bmatrix} 0 & 0 & \frac{1}{J_G \cdot r_{GB}} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{J_D} & 0 & \frac{r_{AW}}{J_D} & 0 \\ \frac{K_1}{r_{GB}} & -K_1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{b}{M \cdot K_{AC}^2} & \frac{1}{M} & -\frac{b}{M \cdot K_{AC}} \\ 0 & K_2 \cdot r_{AW} & 0 & -K_2 & 0 & 0 \\ 0 & 0 & 0 & \frac{b}{M_L \cdot K_{AC}} & 0 & \frac{b}{M_L} \end{bmatrix};$$

The input distribution matrix is

$$B = \begin{bmatrix} \frac{1}{J_G} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix};$$

The measurement gain matrix is $C = [0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 1]$; and $D = [0]$.

Simulations can now be performed as the state space model which completely defines the system has been derived from the Linear Graph model. The simulations allow the design engineer to determine the system response to external inputs without the need to build a real system or make changes to an existing system. The developed analytical model of the IIB conveying system is validated by comparing simulation results with the actual results obtained from the experimental setup. MATLAB® based Simscape™ language is used to code the programs for simulation. Simscape is an extension of Simulink® software which allows modeling of a multi-domain physical system by using blocks that represent physical components and their connections. Simscape™ along with its extensions SimMechanics® and SimElectronics® provide the required blocks for modeling and simulation of the electromechanical conveying system of the IIB. Furthermore, the software allows the development and integration of custom developed blocks.

The model of the conveying system is divided into two subsystems, the "driving" subsystem and the "driven" subsystem. The power source and the gearbox form the driving subsystem whereas the driven subsystem consists of a wheel/axle mechanism coupled to a 3-linkage mechanism. For the purpose of LG modeling, the power source is considered as an ideal torque source formed by an AC induction motor connected to a VDP. However, for the experimental setup, the VDP is replaced by a Variable Frequency Drive (VFD) shown in Fig. 13. The VFD is used to control the four-pole, 1hp, AC motor (Fig. 13a) in closed loop to maintain a constant torque and a constant speed. The AC motor is directly coupled to a 3-stage helical gearbox with a gear ratio of 106.58:1. The Simscape model for the driving subsystem is shown in Fig. 14.

Ideal torque sensors are connected to the input shaft (LHS) and the output shaft (RHS) to verify that proper relationships between elements have been incorporated into the model. Both sensors are connected to a Scope to monitor whether the gearbox acts as a constant speed reducer.

On applying a sine wave input signal (simulating a no-load input speed of 1800 rpm) the ideal gearbox exerts a peak output speed of 16.8 rpm as shown in Fig. 15. Similarly, with no-load and the same input sine wave, the peak torque increases from 0.1584 to 16.88 Nm.

The 'driven' subsystem, which consists of a wheel/axle mechanism coupled to a 3-linkage mechanism, converts the rotational motion of the AC motor into the translational push–pull movement of the conveying system.

The linkage AE is the final element of the mechanism that transmits the force required to move the sliding copper bars. These copper bars have several indexing pins which fold only in one direction thereby allowing the fish to be transported with a linear intermittent movement through the table of the Iron Butcher. The model for this section is shown in Fig. 16. The linkages are represented by blocks called "bodies" (shaded) which include information such as mass, moment of inertia tensor, the location of the center of gravity, and the physical location with respect to the global or local coordinates in the system. The bodies are conveying bars, linkage AE, linkage AB, linkage CD, and disk DM.

Every block is connected to a joint block (revolute joints 1, 2, 3, 4, and 5, and a prismatic joint) which represents the degrees of freedom (DOF) of the body relative to another body, or relative to the global reference in the system. Similar to the Simscape model, the SimMechanics model is able to incorporate joint sensors and scopes for visualizing the responses corresponding to different inputs. After determining the parameters of every component in the system, and specifying them in the model along with their units, the simulation has shown the following results for a constant input speed of 1800 rpm:

Fig. 13. Driving subsystem; (a) AC motor and gearbox, (b) VFD.

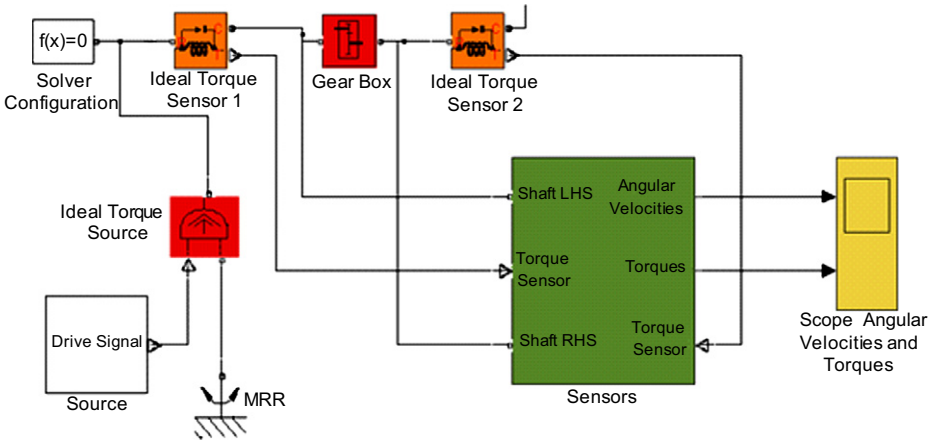

Fig. 14. Simscape model of the driving section.

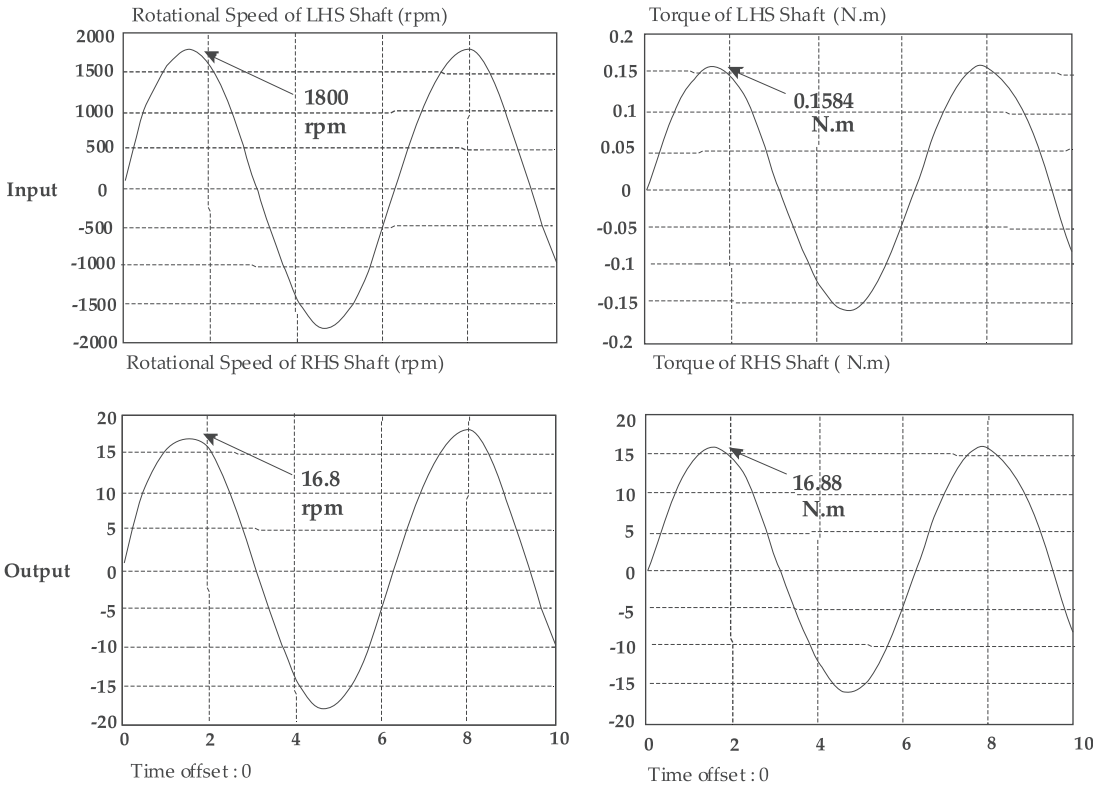

Fig. 15. Gearbox input and output signals.

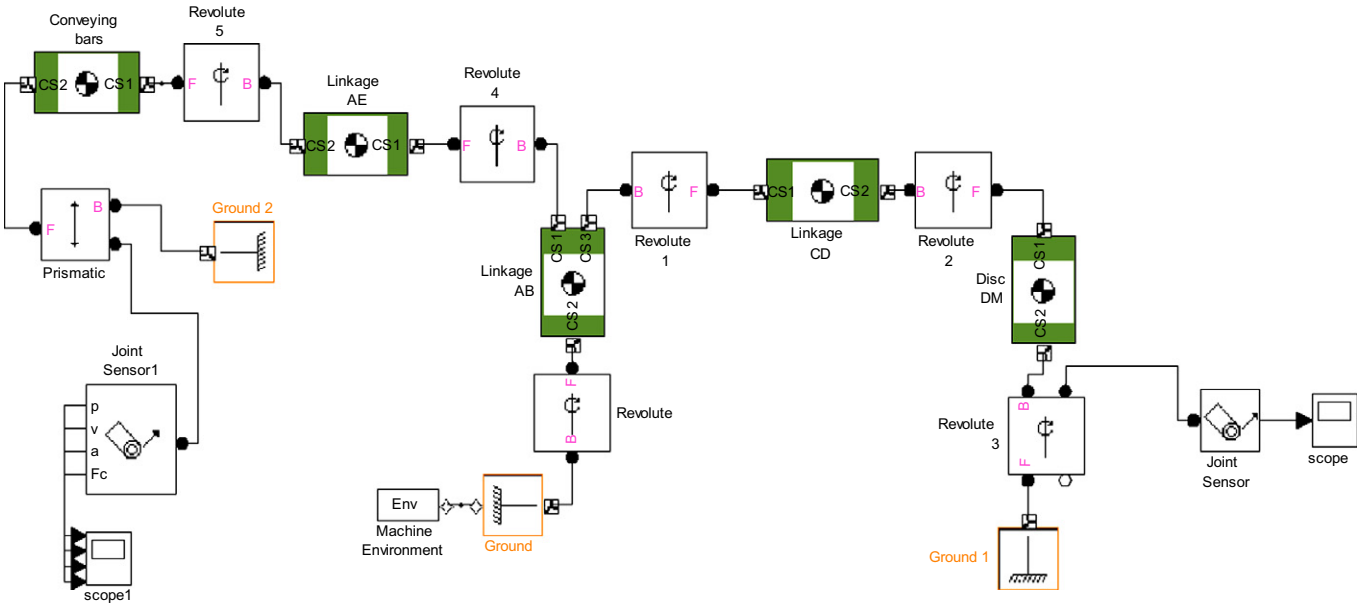

Fig. 16. SimMechanics model of the driven section.

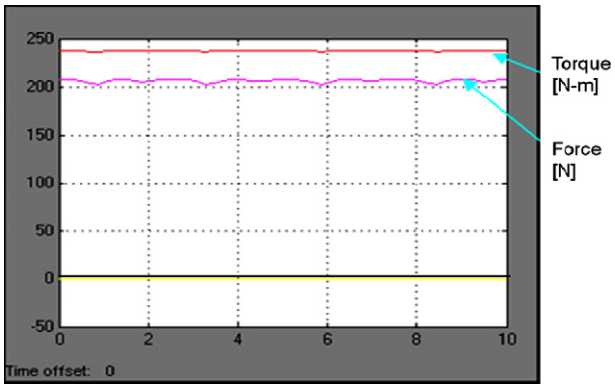

Fig. 17. Reaction torque and reaction force.

- Simulation time = 20 s.
- Average distance traveled by the indexing pins per cycle = 30.3 cm.
- Average time per cycle = 2.53 s.

Fig. 17 shows the reaction torque and the reaction force measured in the copper bars with respect to the linkage AE.

Although the calculated parameters are realistic, several assumptions have been made. To verify the correctness of the model, the results are compared with experimental data. For example, the measured distance the indexing pins travel on the Iron Butcher is approximately 28.5 cm. Also the average time to complete a cycle is 3.05 s. The existence of differences is understandable as the model has not taken into consideration several aspects such as the losses due to friction between the copper bars and the support on which they rest.

4. Fault detection and design improvement

The machine health monitoring system implemented on the IIB [9] has detected that the quality and the accuracy of the fish head removal process have degraded as a result of the conveyor speed being slow. The DES has determined, based on the knowledge gathered over a period of time that the corresponding shortcoming is in the gearbox subsystem due to a design weakness. Therefore, the design objective is to find the proper gearbox with the correct gearbox ratio that yields the required conveyor speed. At this stage the design evolution process is activated and an embryo model is generated for the system. As the weakness is in the gearbox subsystem, the components corresponding to the gearbox are removed from the LG model and two modifiable sites are introduced instead to initiate model evolution, as shown in Fig. 18.

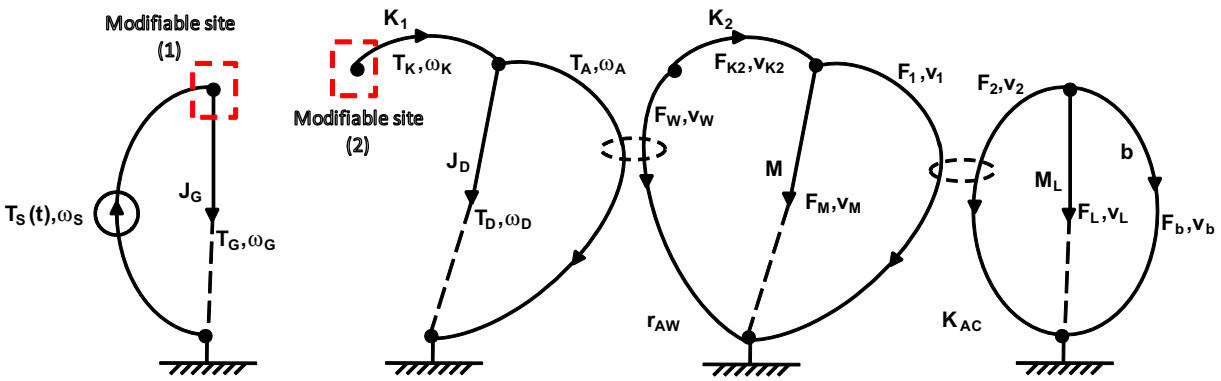

Fig. 18. Linear Graph Embryo model with two modifiable sites.

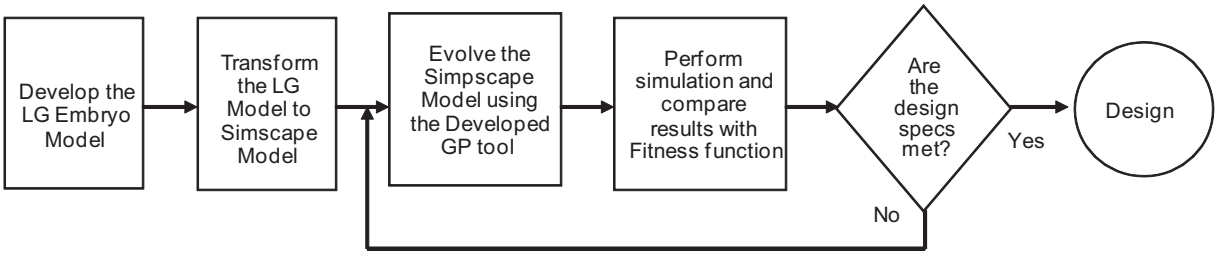

Fig. 19. The evolutionary, simulation, and evaluation processes.

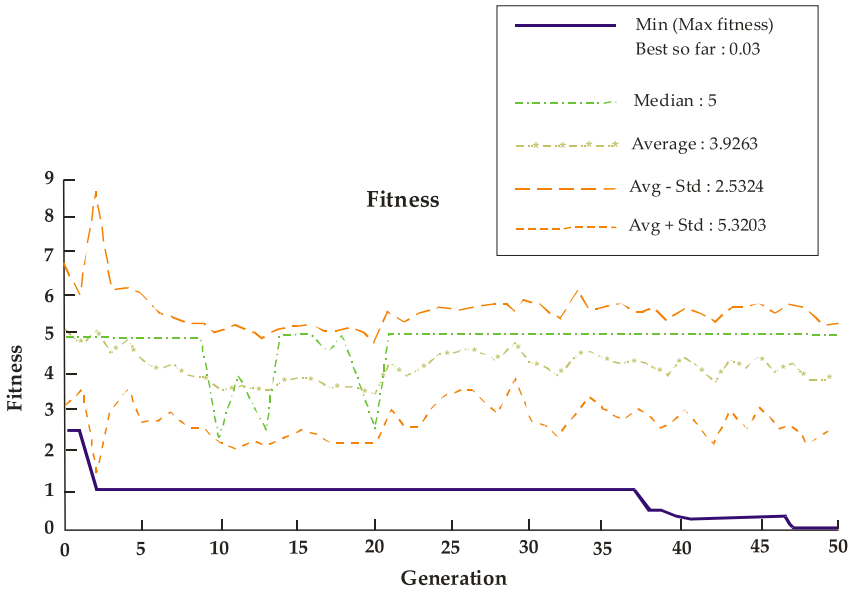

Fig. 20. Fitness plot.

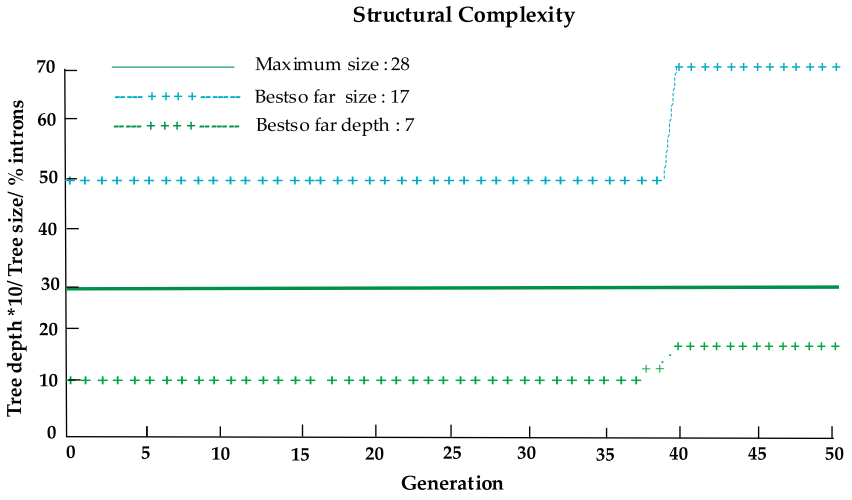

Fig. 21. Complexity plot.

The embryo model is then converted into a Simscape model for the purpose of design evolution using the developed tool as shown in Fig. 19. The performance of each individual evolved through the GP tool is compared with a specified fitness function which corresponds to a design specification of 30 cm displacement of indexing pins in 2.5 s. The input is kept constant at a speed of 1800 rpm. In addition to comparing the performance using the fitness function, factors such as complexity of the evolved individual and the variation of the component parameters from the current values are also considered.

The evolution is carried out for 50 generations with a population size of 100. The best GP tree individual is found in the generation 47 and it has a fitness value of 0.0300. The best GP tree corresponds to a design of a conveying system with a gearbox ratio of 106.58:1 compared to the existing gearbox with a transmission ratio of 147.92:1. The existing gearbox has an average time per cy-

cle of 3.04 s with an average distance traveled by the indexing pins of 30.5 cm. The operators of the IIB would have had to increase the motor speed to compensate for the 0.54 s delay in every cycle and as a result the gearbox would have been functioning outside its design specifications causing it to degrade. The new gearbox with a transmission ratio of 106.58:1 produces an average time per cycle of 2.53 s which satisfies the original design specification of 2.5 s per cycle. Fig. 20 presents the fitness of the best individual found, as well as the evolution median, average, and average ± standard deviation values of fitness.

Fig. 21 presents the evolution of the tree depth and size during the run. Also the plot shows the values of the best individual found.

5. Conclusions

This paper developed a design evolution framework to automate the evaluation and improvement of the design of an existing mechatronic system. The developed system framework integrated a Machine Health Monitoring System (MHMS), a Design Expert System (DES) with an Evolutionary Design Optimization System (EDOS) to obtain a design solution that satisfies the design specifications. A methodology was presented to systematically model mechatronic system designs using Linear Graphs (LGs). A methodology was developed to evolve an LG model of a mechatronic system using Genetic Programming (GP) until a design model that satisfied the required level of performance was found. Knowledge obtained from the MHMS and design experts was captured by the DES and used to guide the evolutionary process and to improve its efficiency. The developed methodology was applied to obtain an LG model for the electromechanical conveying system of an industrial fish processing machine and the corresponding state-space model. The developed design evolution framework used this model to execute the evolutionary process an redesign the conveying system after a weakness in its gearbox was detected by the MHMS.

Acknowledgments

This work has been supported by research grants from the Natural Sciences and Engineering Research Council (NSERC) of Canada, the Canada Foundation for Innovation (CFI), the British Columbia Knowledge Development Fund (BCKDF), and the Canada Research Chair in Mechatronics and Industrial Automation held by C.W. de Silva.

References

[1] Gamage LB, de Silva CW. A system framework with on-line monitoring and evaluation for design evolution of engineering systems. J Comput Inform Sci Eng 2010;10:1–6.
[2] Suh NP. Axiomatic design: advances and applications. MIT-Pappalardo series in mechanical engineering. New York, NY: Oxford University Press; 2001.
[3] de Silva CW. Mechatronics—a foundation course. Taylor & Francis, Boca Raton, FL: CRC Press; 2010.
[4] Behbahani S, de Silva CW. Mechatronic modeling and design, mechatronic systems – devices, design, control, operation, and monitoring. Boca Raton, FL: CRC Press, Taylor & Francis; 2007.
[5] Seo K, Fan Z, Hu J, Goodman ED, Rosenberg RC. Toward a unified and automated design methodology for multi domain dynamic systems using bond graphs and genetic programming. Mechatronics 2003;13:851–85.
[6] Koza JR, Bennett FH, Andre D, Keane MA. Synthesis of topology and sizing of analog electrical circuits by means of genetic programming. Comput Methods Appl Mech Eng 2000;186:459–82.
[7] Karray F, de Silva CW. Soft computing and intelligent systems design. New York, NY: Addison-Wesley; 2004.
[8] Hu J, Goodman E, Rosenberg R. Topological search in automated mechatronic system synthesis using bond graphs and genetic programming. In: Proc American control conference. Boston, Massachusetts; 2004. p. 5628–34.
[9] Lang H, de Silva CW. Fault diagnosis of an industrial machine through sensor fusion. Int J Inform Acquis 2008;5:93–110.
[10] Raman S, de Silva CW. Classifier design for sensor-fault tolerant condition monitoring in an industrial machine. In: Proc ASME dynamic systems and control conference, Hollywood, CA; 2009. p. 1–7.
[11] de Silva CW. Mechatronics – an integrated approach. Taylor & Francis, Boca Raton, FL: CRC Press; 2005.
[12] de Silva CW. Modeling and control of engineering systems. Taylor & Francis, Boca Raton, FL: CRC Press; 2009.
[13] de Silva CW. Sensory information acquisition for monitoring and control of intelligent mechatronic systems. Int J Inform Acquis 2004;1:89–99.
[14] Behbahani S, de Silva CW. Mechatronic design quotient as the basis of a new multicriteria mechatronic design methodology. IEEE-ASME Trans Mech 2007;12:227–32.
[15] Holland JH. Adaptation in natural and artificial systems: an introductory analysis with applications to biology. Control and artificial intelligence. Ann Arbor, MI: University of Michigan Press; 1975.
[16] Koza JR. Genetic programming: on the programming of computers by means of natural selection. Cambridge, MA: The MIT Press; 1992.
[17] Razavi B, de Silva CW. Condition monitoring in a hydraulic system of an industrial machine using unscented Kalman filter. Int J Inform Acquis 2010;7:177–92.