A Sequential Linear Program for the Automatic Loading of Vessels

Sérgio Alvares R. de S. Maffra, *Tecgraf/PUC-Rio*

Luiz Cristovão Gomes Coelho, *Tecgraf/PUC-Rio*

Guilherme Tavares Malizia Alves, *Tecgraf/PUC-Rio*

Mauro Costa de Oliveira, *Petrobras/Cenpes*

Carlos Gomes Jordani, *Petrobras/Cenpes*

ABSTRACT

Large ships and platforms are kept in equilibrium by controlling the amount of mass in its tanks (ballast, oil, diesel, etc) as a way to compensate the disequilibrium caused by external forces, by its cargo or by its own weight. In this paper we present a sequential linear program that calculates the amount of liquid in each tank necessary to put the vessel at any desired position while minimizing the amount of liquid displacement, the structural stress or the free surface of all tanks selected for change.

Keywords: *linear programming, vessel stability, automatic loading*

1. INTRODUCTION

Sequential linear programs solve nonlinear problems by a sequence of linear approximations that are solved using linear programming (Bertsimas & Tsitsiklis, 1997) Generally speaking, these algorithms are iterative procedures that, at each step, try to guess the solution of the nonlinear problem using the solutions provided by the linear approximations until convergence is achieved. In this problem the nonlinearity arises in the computation of the center of gravity of the tanks, which are allowed to have an arbitrary geometric form, position and orientation. So, the longitudinal and transversal center of gravity values (LCG and TCG) may vary during the filling process. The linear approximations consist in considering each tank as a punctual load so that the equilibrium of the vessel can be easily expressed as the constraints of a linear program, responsible for computing the concentrated mass in each tank. After computing these masses, the algorithm computes the actual position of the center of gravity of each tank, using the correct geometry. Convergence is achieved when the position of the punctual loads and the center of gravity of the tanks do not vary significantly. Using the minimization of the variation of mass at each punctual load as the objective function of the linear program, we guarantee that the solutions obtained correspond to the minimum displacement of liquids.

Our algorithm incorporates an optional heuristic to minimize the structural stress of the vessel, which consists in trying to achieve an even distribution of liquids among the cargo, ballast and oil tanks, assuming that the loading distribution can be similar to the buoyancy. This algorithm has been implemented in the

Sstab Program (Coelho et al., 2003) and has proved its efficiency in the design of ships and oil platforms (Campos Basin, Rio de Janeiro, Brasil) and also in the design of emergency plans performed by the Classification Societies ABS, DNV, LR, and BV.

Our algorithm also incorporates a second optional heuristic to minimize the free surface of the tanks vessel, which consists in trying to concentrate a maximum amount of liquids in the smaller number of tanks possible.

2. LINEAR PROGRAMMING

As mentioned in the Introduction, the algorithm proposed uses linear programming in order to compute an equilibrium configuration for a vessel. Generally speaking, linear programs (an instance of a linear programming problem) have two components: a set of linear constraints and a cost function that can be maximized or minimized. A general formulation for linear programs is shown in Equations 1 to 6.

Minimize / Maximize

$$C^t X \qquad (1)$$

Subject to

$$A_i^t X \geq b_i \qquad i \in M_1 \qquad (2)$$

$$A_i^t X \leq b_i \qquad i \in M_2 \qquad (3)$$

$$A_i^t X = b_i \qquad i \in M_3 \qquad (4)$$

$$x_j \geq 0 \qquad j \in N_1 \qquad (5)$$

$$x_j \leq 0 \qquad j \in N_2 \qquad (6)$$

In the figure, uppercase letters represent vectors and lowercase letters represent scalar values. X represents the vector containing all the variables that are being determined by the program and x represents a single variable. Different weights (C) can be assigned to each variable in the cost function (Equation 1). The different kinds of constraints are also shown (in light gray). Notice that equality and inequality constraints are acceptable. The first three sets of constraints (M_1, M_2 and M_3) contain a vector A and a scalar value b, which represent the data associated with a constraint.

A feasible solution for a linear program is a solution that does not violate any of the constraints of the problem. An optimal solution is the feasible solution with the largest or smallest evaluation of the cost function, depending on whether the problem is a minimization or maximization problem.

In the next section the general linear programming problem is specialized for computing the equilibrium of a vessel.

2.1 Linear Program for the Equilibrium of a Vessel

A vessel is in equilibrium when the forces acting on it, and their derived moments, are balanced. That is, buoyancy, weight, external forces and the moments they generate must be balanced. Wind and current forces are examples of external forces that act on a vessel.

Among the forces mentioned above, the only ones that can be controlled are the weight forces resulting from the liquids stored in the tanks of the vessel. Therefore, the variables used in the automatic loading linear program must be related to the amount of liquids in the tanks of the vessel. For flexibility, the variation of volume in each tank was chosen. Given the variables of the problem, the equilibrium constraints and the constraints on the variables themselves can be easily written, as Equations 7 to 12 show.

$$\sum (V_i + \Delta V_i)\rho_i g = B - W \qquad i \in T \qquad (7)$$

$$\sum (V_i + \Delta V_i)\rho_i g x_i = -M_y \qquad i \in T \qquad (8)$$

$$\sum (V_i + \Delta V_i)\rho_i g y_i = -M_x \qquad i \in T \qquad (9)$$

$$V_i + \Delta V_i < V_i^{\max} \qquad i \in T \qquad (10)$$

$$V_i + \Delta V_i > 0 \qquad i \in T \qquad (11)$$

$$\sum (V_c + \Delta V_c) = V_k^{total} \qquad c \in T_k \qquad (12)$$

optional

V_i	Initial volume at tank i
ΔV_i	Variation of volume at tank i
$V_i^{\max}$	Capacity of tank i

ρ_i	Density of the liquid in tank i
G	Gravity constant
x_i, y_i	Coordinates of the center of gravity of tank i
B	Buoyancy force
W	Weight of the vessel
M_x, M_y	Moments resulting from external forces
T	Set of tanks of the vessel
T_k	Set of tanks holding the same kind of liquid (k)
V_k^{total}	Total volume of liquids in tanks belonging to T_k

The first constraint assures a proper balance of forces. In order to be in equilibrium the weight of the liquids in the tanks and the weight of the platform must be equal to buoyancy. The second and third constraints assure the balance of the moments caused by the liquids and the ones caused by external forces. These first three constraints are responsible for maintaining the vessel in equilibrium.

Constraints ten and eleven are relative to the variables of the problem. The former guarantees that no tank receives more liquid than its maximum capacity while the latter guarantees that no tank will hold negative volumes.

The coordinate system to which Equation 2 refers is defined as having the X Axis pointing forward along the longitudinal axis of the model, the Y Axis defines a plane parallel to the sea surface and Z Axis always points upwards. Figure 1 shows a heeled ship and the global Cartesian coordinate system used.

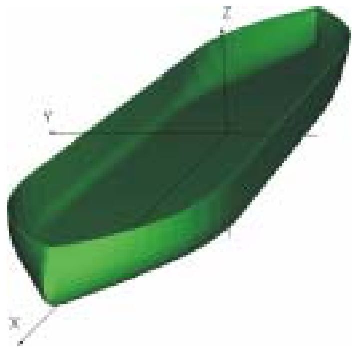

Figure 1- Coordinate System.

It is not uncommon for a vessel to contain tanks holding different kinds of liquids (fresh water, diesel, oil, sea water, etc). In these situations, it is important to guarantee that these different types of liquids are not mixed by the algorithm. That is accomplished by using the constraint expressed in Equation 12. This constraint forces indicates that the tanks belonging to the set Tk hold a determined amount of liquid. Therefore, by creating a set Tk for each type of liquid and using their total volume as the quantity of liquid that they must hold, the mixing of liquids can be easily avoided. This constraint, however, will rarely be used with ballast tanks, as sea water can be disposed of and collected by a vessel. The same is not true, for example, for oil tanks.

When creating an automatic loading program for a vessel, one must decide whether to use or not the sixth constraint. When used, one must also decide how many instances of this constraint will be used and for what types of tanks. Most of the times, this constraint is expected to be used with oil, fresh water and diesel tanks.

2.2 External Forces and Weight Forces

The constraints presented in the previous section account for the existence of external forces and their resulting moments. In our implementation these values are computed directly from a geometric model of the vessel. This model is constructed as a set of individual compartments that define its complete stability model. The compartments can be classified as external (hull body and deck elements) or internal (void spaces, elevators, access trunks, chain lockers, tanks, etc). Each individual compartment is composed of a set of planar faces, which are defined a set of three-dimensional coordinates (Coelho et al, 2003).

Wind forces and their resulting moments are computed, according to IMO's recommendation (IMO, 2001), as a function of the velocity of the wind and of the projected area of the external compartments that are exposed to wind action. The same procedure is used for current forces.

The buoyancy is determined by computing the contribution of all submerged parts of the external compartment. Using the mesh intersection algorithms implemented in the MG library (Coelho et al., 2000), the external compartments are cut into two parts using a plane located at the sea surface position. The center of gravity and the volume of each submerged part are then used to compute the resulting buoyancy force and the center of buoyancy.

The same procedure is used to determine the center of gravity of each tank. Using a procedure similar to a binary search (Cormen, et al., 2001), the position of the free surface plane of each tank is determined based on the volume it is currently holding and also based on its current orientation. Then, after cutting the tanks in two parts (wet and dry surfaces), similar to what was done for the external compartments, the center of gravity can be determined by computing the geometric center of the faces containing liquids.

Figure 2 shows an example of the mesh operations performed on the external compartments and on tanks in order to compute the buoyancy or tank weights and the center of buoyancy or gravity of each compartment (hull or tank). The mesh of the hull is shown in green and the meshes corresponding to tanks, in yellow.

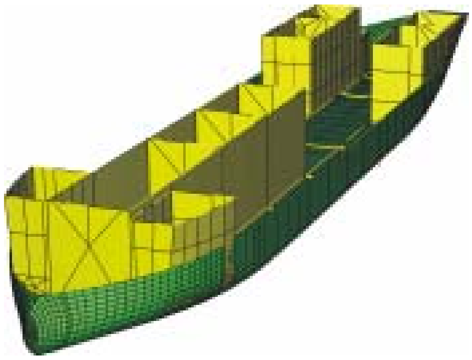

Figure 2- Cutting of compartments.

3. COST FUNCTION

The last component of the linear program created for the automatic loading algorithm that must be described is the cost function. As mentioned before, this function is used to select the best result from set of the feasible solutions of a problem instance. In the automatic loading, the cost function selected minimizes the variation of volume in each tank, as Equation 13 shows.

Another possible cost function is selecting the fastest pumping adjustment, that is, the fastest way of transferring liquids among the available tanks of the floating system. In order to create this cost function data about the pumping capacities of each tank are necessary.

$$\textbf{Minimize} \qquad \sum \Delta V_i \qquad i \in T \qquad (13)$$

4. THE SEQUENTIAL LINEAR PROGRAM

Observing the tanks illustrated in Figure 2, it is easy to see the X and Y coordinates of its center of gravity are not constant, for many tanks at the stern and bow (all of them if the turn of bilge is significant). They are, actually a function of the volume contained in the tank. The linear program developed for the automatic ballast algorithm, however, considers them as constants. In order to consider the correct centers of gravity, the automatic ballast algorithm is implemented as a sequential linear program.

The algorithm proceeds as illustrated in Figure 3. The linear program, as described in Section 2.1 is used to estimate the first variations of volume. These are then used to compute the new centers of gravity of each tank, as described in Section 2.2. Finally, the new centers of gravity (CG) are compared with the previous ones. In case the distance between all of them is smaller than a predefined error tolerance, the algorithm stops and the desired variation of volume in each tank is known. In case the distance between old and new CGs is larger than the error tolerance, the algorithm starts a new iteration, using the new CGs instead of the old ones.

The algorithm also stops after a maximum number of iterations is reached.

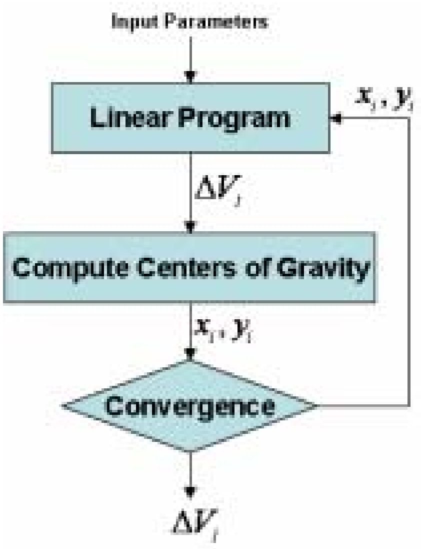

Figure 3- Chart Flow of the loading algorithm.

5. IMPROVING THE RESULTS

The equilibrium of a floating system is an essential requirement for the automatic ballast algorithm. In some cases, however, the consideration of other requirements is also important. Two additional requirements were included in the automatic ballast algorithm: the minimization of the structural stress of the vessel and the minimization of free surface effects of the tanks.

5.1 Minimization of Free Surface Effects

Free surface effects occur when a tank is partially filled with liquids. No free surface effect exists when a tank is empty or completely full. When dealing with semi-submersible platforms or self-elevating units, the minimization of this effect is important to avoid variations in their center of gravity when the vessel heels for some reason.

The same strategy used to deal with the non-linearity of the center of gravity of the tanks can be applied to this problem. That is, the sequential linear program, as depicted in Figure 3 is used as a step of another iterative procedure. This new algorithm minimizes the effects of free surfaces by reducing the number of tanks used in the automatic ballast algorithm as shown in Figure 4. The algorithm begins with an empty set of tanks and at each iteration adds one tank to the set. The equilibrium of the vessel, using the updated set of tanks is then computed. Convergence is achieved when the equilibrium is computed successfully. Otherwise, the algorithm starts a new iteration, including an additional tank and then reapeating the equilibrium computation.

Figure 7 shows an example of a ballast configuration computed using the algorithm presented in this section.

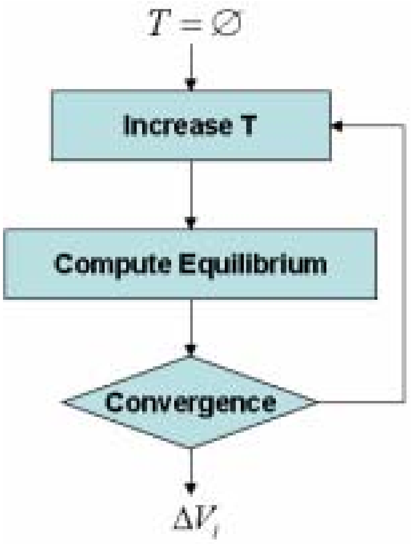

Figure 4- Minimization of free surfaces.

5.2 Minimization of Structural Stress

Some experiments performed with the automatic ballast algorithm have shown that some loading conditions proposed by the algorithm can result in violation of the envelopes of shear-forces or bending moments of large ships. Irregular variations of the shear-forces have also been observed in many applications of the standard algorithm.

Both conditions were observed when the distribution of liquids among the tanks was uneven.

In order to reduce these effects, two more constraints have been introduced in the algorithm. The new constraints control the variation of volume among tanks, trying to create a regular longitudinal distribution of liquids. What, in most cases, will result in a reduction of stresses. Figure 5 shows the buoyancy curve (in blue) for a typical ship vessel. This curve is constant along the parallel body, so, theoretically, if the ship is loaded with a curve that exactly matches the buoyancy curve, the shear and bending effects will be zero.

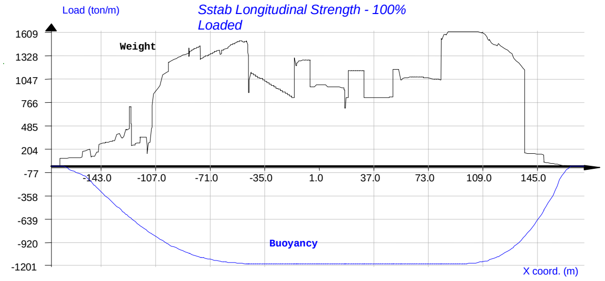

Figure 5- Buoyancy curve of a ship.

The new constraints can be seen in Equations 14 and 15. Two new parameters were introduced: a volume variation tolerance that is used among all tanks (TOLg) and a volume variation tolerance for the neighbouring tanks (TOLn). As Equations 4 and 5 show, the amount of liquid in the tanks will depend on the volume of its neighbouring tanks and of all other tanks.

The optimal value for both TOLg and TOLn is zero, which corresponds to a perfectly even distribution of liquids among the tanks. This value, however, is not acceptable for most cases. The determination of TOLg and TOLn is then performed by using an iterative procedure, as shown in Figure 6.

$$\left| \frac{v_i + \Delta v_i}{V_i^{\max}} - \frac{v_k + \Delta v_k}{V_k^{\max}} \right| < TOL \quad i \in T, k \in T \quad (14)$$
$$i \neq k$$

$$\left| \frac{v_i + \Delta v_i}{V_i^{\max}} - \frac{v_j + \Delta v_j}{V_j^{\max}} \right| < TOL \quad i \in T, j \in N \quad (15)$$

TOL_g	Global volume variation tolerance
TOL_n	Neighbor volume variation tolerance
N_i	Set of neighbor tanks of tank i

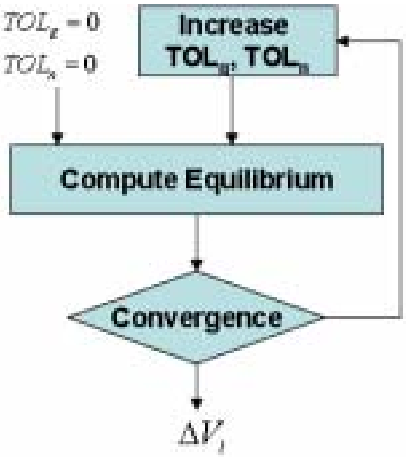

Figure 6- Determination of TOLg and TOLn.

6. RESULTS

We selected three examples to show the possibilities of the automatic loading algorithm. The first one shows a semi-submersible platform with 24.000 tons of ballast capacity with an operational draft of 23.10 meters. The total light weight is 25.600 tons and the CG of this light weight is 2.4 meters forward and 1.1 meters starboard. After selecting all ballast tanks (initially empty) to balance the model in even keel at the operational draft, the results obtained with the minimization with the free-surface parameter activated are shown by Table 1

Figure 7 shows the tank loading defined by the algorithm. The compartments containing liquid are shown in solid blue. All other compartments are shown with transparency.

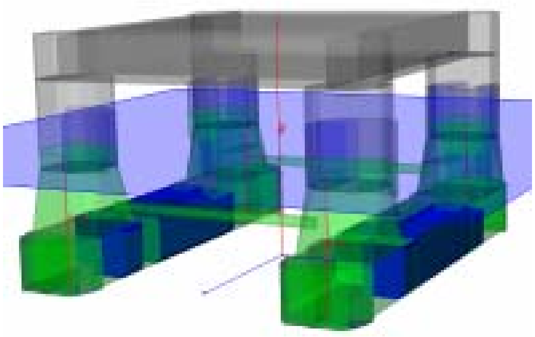

Figure 7- Ballast loading after algorithm.

Table 1- Tank loading after algorithm.

Name	Capacity (t)	Final Weight (t)	Final Filling (%)
BT_PS_FWD_09	441.67	441.67	100
BT_SB_FWD_21	441.67	441.67	100
BT_SB_FWD_22	427.14	427.14	100
BT_PS_FWD_11	438.71	438.71	100
BT_SB_FWD_19	438.71	438.71	100
BT_PS_FWD_13	438.71	438.71	100
BT_SB_FWD_17	438.71	438.71	100
BT_PS_FWD_14	622.88	622.88	100
BT_SB_FWD_18	622.88	622.88	100
BT_PS_MDS_15	438.71	438.71	100
BT_SB_MDS_15	438.71	438.71	100
BT_PS_MDS_16	819.31	646.82	79
BT_SB_MDS_16	819.31	819.31	100
BT_PS_AFT_09	441.67	352.32	80
BT_PS_AFT_11	438.71	438.71	100
BT_PS_AFT_11	438.71	438.71	100
BT_PS_MDS_15b	438.71	438.71	100
BT_SB_AFT_19	438.71	438.71	100
BT_SB_AFT_18	622.88	559.64	90
BT_PS_AFT_11	438.71	438.71	100
BT_PS_MDS_15b	438.71	438.71	100

It possible to see in Figure 7 that the most significative trim and heel tanks were not used. This is because of the sorting procedure applied to the selected set of tanks that was performed based on the heel and trim moment capacities and, with the minimization of free-surface parameter selected, those tanks are the last to be used.

The second example consists of a FPSO with the minimization of longitudinal stresses parameter selected. In this example the idea is to maintain the total weight (light-weight + ballast + cargo + oil tanks) but to redistribute the cargo in order to minimize the shear and bending longitudinal forces.

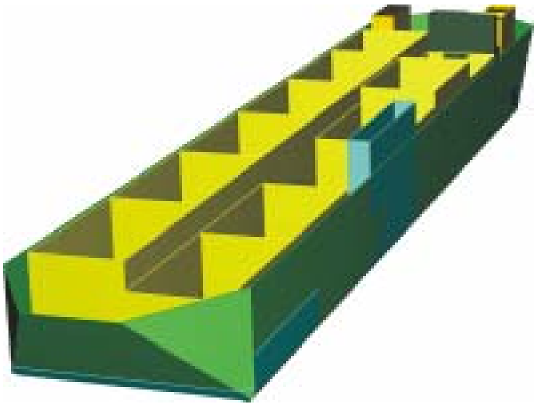

Figure 8 – Ship loaded with minimum shear.

Figure 8 shows the final loading of the tanks. As in the other images captured from Sstab, the meshes in green show the external hull. Ballast tanks are shown in solid blue and oil tanks in solid yellow.

Figures 9 and 10 show the graphs of shear forces and bending moments, respectively.

The last example shows an emergency situation where the automatic loading is used to restore the position of a platform after a damage occurrence. Figure 11 shows the model balanced after the damage. Damaged tanks are shown in solid red.

As shown in Figure 12, a new tank loading was defined by the algorithm to restore the platform to the even keel position before the recovering of the tanks.

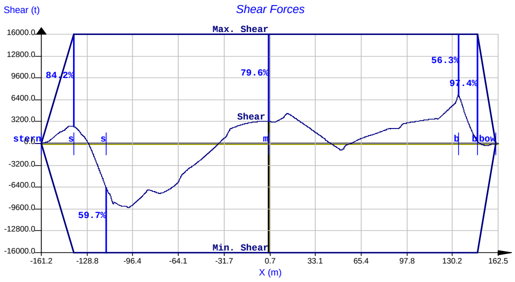

Figure 9- Shear forces of Figure 8.

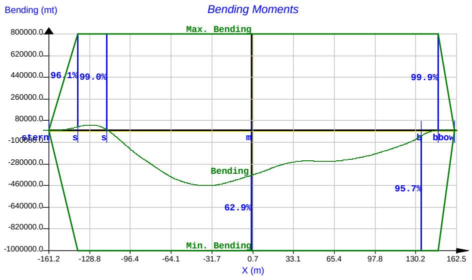

Figure 10- Bending forces of Figure 8.

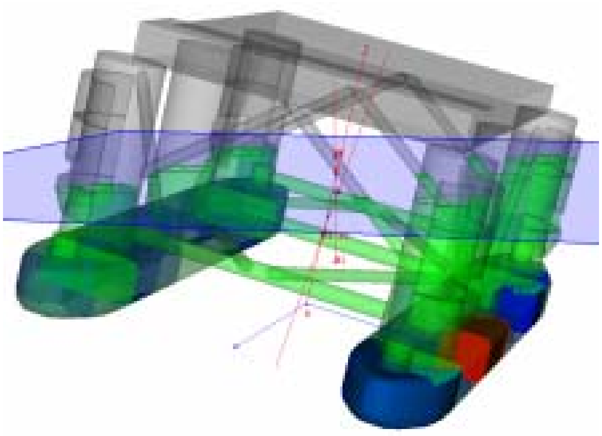

Figure 11- Inclination due to damage.

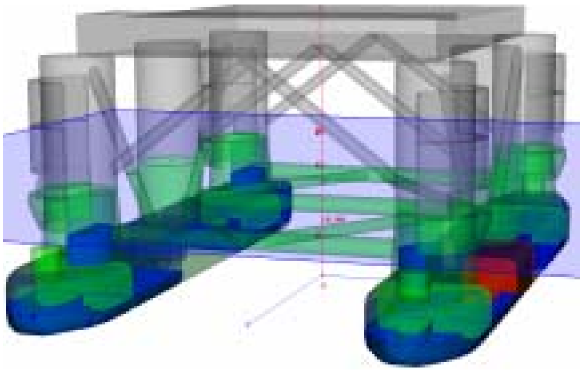

Figure 12- New loading restores even-keel.

7. CONCLUSIONS

An algorithm to balance the loading of tanks was successfully defined and implemented in Program Sstab. This algorithm is responsible for defining the operational conditions of semi-submersible oil platforms and FPSOs that are being designed by Petrobras.

The algorithm is fast enough to run in standard PCs. The tests were made in a Pentium IV with 2.8 GHz CPU capacity and 512 Mb of RAM and, in all three examples, Sstab took about a minute to complete the new loading definitions.

One of the major problems of using a linear programming library to solve this non linear problem is that there are no partial results for the loading condition. The user may want an answer like *how much could you do*. We are also working on this problem, using the same linear programming library. As an alternative, some non-linear library could be tested.

We are improving the data of the program to include the pumping properties and the costs of each operation, what will enable the implementation of the cost optimization.

8. ACKNOWLEDGMENTS

The authors would like to thank Petrobras, CENPES and Tecgraf/PUC-Rio for their support and for the opportunity of working in this project.

9. REFERENCES

Bertsimas, D. and Tsitsiklis, J., 1997, "Introduction to Linear Optimization,," Athenas Scientific.

Coelho, L.C.G., Figueiredo, L.H. and Gattass, M., 2000, "Intersecting and trimming parametric meshes on finite-element shells". International Journal for Numerical Methods in Engineering, Vol. 47, No. 4, pp. 777-800.

Coelho, L.C.G., Jordani, C.G., Oliveira, M.C. and Masetti, I.Q., 2003, "Equilibrium, Ballast Control and Free-Surface Effect Computations Using the Sstab System", Proceedings of the 8th International Conference on the Stability of Ships and

Ocean Vehicles, pp. 377-389.

Cormen, T.H., Leiserson, C.E., Rivest, R.L. and Stein, C., 2001, "Introduction to Algorithms," MIT Press and McGraw-Hill.

IMO, 2001, "Code for the Construction and Equipment of Mobile Offshore Drilling Units."