Proceedings of the International Conference on Simulation and Multimedia in Engineering Education (WMC 2001).
January 2001. Phoenix [SCS]. ISBN 1-56555-224-5

TRAVIS
an engineering tool to animate and validate AGV-systems

Mark B. Duinkerken
Mechanical Engineering and Marine Technology
Delft University of Technology
Mekelweg 2, 2628 CD Delft, The Netherlands
e-mail : m.b.duinkerken@wbmt.tudelft.nl

Marc J. Terstegge
Logistics Research
Europe Combined Terminals
Rotterdam, The Netherlands
e-mail : Marc.Terstegge@ect.nl

Keywords: Animation, Control systems, Discrete simulation, Transportation.

Abstract

Most of the AGV systems currently in operation are either relatively small in size (less then 10 AGVs), or control is simplified (fixed routes, closed loop systems). However in the near future, larger, more complex systems are expected.

Engineering education at the transportation department of the university make students aware of the relevant design and control parameters. Simulation models for both the logistic and physical processes play a key role.

In this paper, our work on visualization of the physical process, and on algorithms to control the physical integrity of the system, is presented.

INTRODUCTION

Every year containerized transport shows a large growth worldwide. As a response to the growing demand for transportation and in order to reduce labor costs, Europe Combined Terminals (ECT) in Rotterdam introduced a high degree of automation on the terminals. The Delta Sealand terminal (1993) was the first fully automated container terminal.

An automated container terminal uses automated guided vehicles (AGVs) for the intra-terminal transport of containers. The operational control requires an advanced AGV operating system. Such a system must be able to :

- minimize the variance in transport times
- maximize terminal capacity fi. by dynamic routing
- detect and prevent collision between AGVs, and with infrastructural elements
- detect and prevent deadlock situations
- allow topology changes.

Specific for the container terminal is the size of an AGV, 20 meters (compared to 1 or 2 m. for most industrial AGVs), and the costs of the infrastructure, which makes it essential to optimize the throughput with minimal land-use.

The former Dutch Center for Transport Technology (nowadays named Connekt) has started research on the design of a new generation of container terminals. These terminals have to be equipped for handling the soon to be expected Jumbo Container Vessels with a capacity of 8000 TEU (Twenty-feet Equivalent Units) and more. The second phase of this research project will start soon.

TRAFFIC CONTROL

Research at the Delft University of Technology resulted in the AGV operating system TRACES (TRAffic Control Engineering System). This system has been used to design and evaluate existing and new terminal layouts. [Evers, Duinkerken]

Conventional traffic control is directly linked with the traffic infrastructure by traffic lights and situational traffic rules. In TRACES traffic control takes place entirely within a computer model, where, within some margin, synchronisation is preserved between the progress of the physical processes (i.e. the moves of the vehicles) and the evolution of the state of the computer model. The TRACES approach was tested using simulation.

In the set-up of TRACES, vehicles drive along routes, coded in scripts. A route is the path that a vehicle must travel in order to carry out a transport order. Nodes and corridors are used to define a path. A node is an indication of a position in the infrastructure, a corridor is a predefined link between two nodes. In the simplest case this link is a straight line, but it may also be used to describe more complex vehicle movements such as 90° bends, S-bends and lateral movement. Figure 1 shows an example of a simple layout.

In order to avoid deadlocks and vehicle collisions and to increase vehicle flow, semaphores [Dijkstra, Ben-Ari] are used to virtually control traffic flow. A script describes the physical route between start- and endpoint, and indicates when specific semaphores must be claimed and released. A number of alternative physical routes may be defined in one script, in which dynamically, during the execution of the script, a specific route is chosen on the basis of the status of the semaphores.

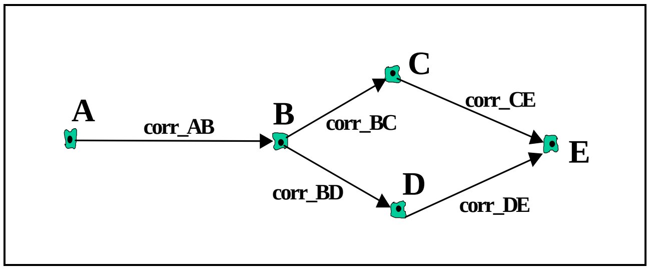

Figure 1 Example of a simple layout with 5 nodes and 5 corridors

As an example, a script is defined in the layout shown in Figure 1. Let S1, S2 and S3 be semaphores with capacity 1. The script, given in Figure 2, describes the route between the nodes A and E. Depending upon the availability of a semaphore (S2), the AGV executing this script chooses between two subscripts, defining different physical routes.

Vehicles are assumed to be able to maintain the appropriate distance from vehicles driving in front of them. This implies that, in principle, vehicles may drive in line in the same ‘control area’. They must also be equipped with the facilities needed for independent data-processing.

The use of straight lines connecting nodes in TRACES offers the possibility for fast design and testing of terminal layouts. However, this causes also that the simulation models lack realistic turning and acceleration behavior of its vehicles. This behavior has to be taken into account to prove that the AGVs in the simulation model actually drive collision free.

```
Script syntax      Description

check A            position is node A
ask S1             wait semaphore S1
do corr_AB         drive corridor from A to B
free S1            signal semaphore S1
check B            position is node B
select             select subscript :
  if S2
    ask S2           if S2 available, then go
    do corr_BC       from B to E via node C,
    do corr_CE
    free S2
  endif
  else               else go via node D
    ask S3
    do corr_BD
    do corr_DE
    free S3
  endelse
endselect
check E            position is node E
```

Figure 2 Example of a script

TRANSLATING THE LOGISTIC MOVES OF AGV'S TO ACCURATE PHYSICAL MOVEMENTS

To increase the reliability of current simulations, the software module TRAVIS (TRACES Visualization) has been developed. This module can be linked with the existing simulations. TRAVIS offers a new visualization for TRACES. In this visualization, realistic turning behavior and acceleration behavior have been added to the square tracks followed by the AGVs in the current simulation, which results in AGV movements that are more natural.

Internally the simulation calculates the AGV positions and orientations using square paths with abrupt changes of orientation. This simplification has great advantages in describing the AGV movements as logistic processes. In reality however, AGVs cannot change their orientation from 0° to 90° in zero time. For the actual driving movements, these square paths are therefore projected onto realistic bend curves. This is shown in Figure 3, where the AGV drives a square ‘bend’ from point $(x_0-\Delta x,y_0)$ via point (x_0,y_0) to point $(x_0,y_0+\Delta y)$.

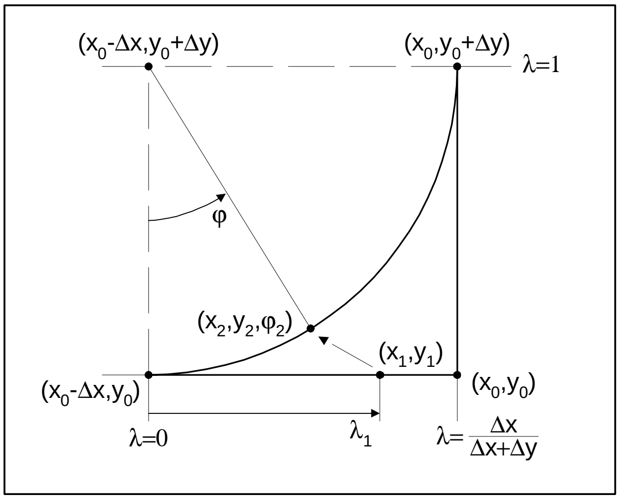

Figure 3 Projection of a route point on a square path onto a realistic bend curve

This projection uses a parameterization of the square path, described by parameter λ, with λ =0 at the beginning of the bend in point (x_0-Δx,y_0) and λ =1 at the end in point (x_0,y_0+Δy). A point (x_1,y_1) on the square 'bend' is described by:

$$\begin{pmatrix} x_1 \\ y_1 \end{pmatrix} = \begin{cases} \begin{pmatrix} x_0-\Delta x \\ y_0 \end{pmatrix} + \dfrac{\lambda}{\alpha}\begin{pmatrix} \Delta x \\ 0 \end{pmatrix} & , \quad 0 \le \lambda \le \alpha \\ \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} + \dfrac{\lambda-\alpha}{1-\alpha}\begin{pmatrix} 0 \\ \Delta y \end{pmatrix} & , \quad \alpha < \lambda \le 1 \end{cases}$$

Where α is the symmetry factor of the bend:

$$\alpha = \frac{\Delta x}{\Delta x + \Delta y}$$

For symmetrical bends, where Δx = Δy, α = ½.

For a given point (x_1,y_1), λ can be calculated by:

$$\lambda = \begin{cases} \dfrac{\alpha(x_1 - x_0 + \Delta x)}{\Delta x} & , \quad x_0 - \Delta x \le x_1 \le x_0 \\ \alpha + \dfrac{(1-\alpha)(y_1 - y_0)}{\Delta y} & , \quad y_0 < y_1 < y_0 + \Delta y \end{cases}$$

Using this parameterization, any point (x_1,y_1) on the square track can be described by parameter λ and then projected onto a point (x_2,y_2,φ_2) on the actual driven bend shape. For example, if the AGV drives a ellipse-shape bend, the projection is:

$$\begin{pmatrix} x_2 \\ y_2 \\ \varphi_2 \end{pmatrix} = \begin{pmatrix} x_0 - \Delta x \\ y_0 + \Delta y \\ 0 \end{pmatrix} + \begin{pmatrix} \Delta x \cdot \sin\left(\frac{1}{2}\pi\lambda\right) \\ -\Delta y \cdot \cos\left(\frac{1}{2}\pi\lambda\right) \\ \frac{1}{2}\pi\lambda \end{pmatrix}$$

In TRAVIS however, table bends are used. These are bends that are described by a table of coordinates and orientations:

x	**y**	**φ**
x_1	y_1	φ_1
x_2	y_2	φ_3
x_3	y_3	φ_3
...	...	...
x_n	y_n	φ_n

These table bends, which can be read from a file, can describe every desired bend curve, including S-shaped bends (although a different parameterization for (x_1,y_1)(λ) is needed). There is no need for any parameterized description of the actual curve, like in the example above.

Before the table can be used for projection, an extra column must be added, which defines λ_i for a bend point (x_i,y_i,φ_i):

λ	**x**	**y**	**φ**
λ_1	x_1	y_1	φ_1
λ_2	x_2	y_2	φ_3
λ_3	x_3	y_3	φ_3
...	...	...	...
λ_n	x_n	y_n	φ_n

Parameter λ_i , which can be seen as the finished fraction of the total table bend in point (x_i,y_i,φ_i), can be calculated as follows:

$$\begin{cases} \lambda_1 = 0 \\ \lambda_i = \lambda_{i-1} + \dfrac{\sqrt{(x_i - x_{i-1})^2 + (y_i - y_{i-1})^2}}{\sum_{j=1}^{n-1}\sqrt{(x_{j+1} - x_j)^2 + (y_{j+1} - y_j)^2}} & , \quad 2 \le i \le (n-1) \\ \lambda_n = 1 \end{cases}$$

The projection of point (x_1,y_1) on the square path onto point (x_2,y_2,φ_2) on the table bend can be calculated using interpolation:

$$\begin{pmatrix} x_2 \\ y_2 \\ \varphi_2 \end{pmatrix} = \begin{pmatrix} x_i \\ y_i \\ \varphi_i \end{pmatrix} + \frac{\lambda - \lambda_i}{\lambda_{i+1} - \lambda_i}\begin{pmatrix} x_{i+1} - x_i \\ y_{i+1} - y_i \\ \varphi_{i+1} - \varphi_i \end{pmatrix} \quad , \quad \lambda_i < \lambda < \lambda_{i+1}$$

In general, the total driving distance in the TRACES model and the projection in TRAVIS will differ. This effect can be compensated for in TRAVIS using a different speedprofile. Furthermore, because the logistic model does compensate for the acceleration and deceleration behaviour of an AGV, the TRAVIS model can calculate the exact position of an AGV at each moment. This allows an accurate animation of the traffic movements.

ALGORITHMS FOR COLLISION DETECTION AND DISTANCE MONITORING

Realistic AGV movements make collision detection and distance monitoring possible. A set of mathematical algorithms is developed for this task and implemented in TRAVIS. TRAVIS is capable of detecting collisions between AGVs. Furthermore, a safety distance can be defined and TRAVIS will notice when two AGVs are violating this distance.

The algorithm for collision detection uses AGV shapes described as polygons, as shown in Figure 4. A collision is detected if the two polygons overlap. This is the case if any of the segments of polygon 1 intersects with any of segments of polygon 2. For two polygons, respectively consisting of n and m segments, the pseudo-code of the algorithm is formulated as:

```
overlap=false;
i=1;
while (i ≤ m)
{
  j=1;
  while (j ≤ n)
  {
    if (Intersect(Segment i, Segment j))
    {
      overlap=true;
    }
    j=j+1;
  }
  i=i+1;
}
```

The heart of this algorithm is the function ‘Intersect’, which determines if two line segments intersect.

Distance monitoring is achieved in a similar manner. Instead of checking the intersection of two segments, the distance between two segments is calculated. The distance between two AGVs is the minimum of all the distances between their segments.

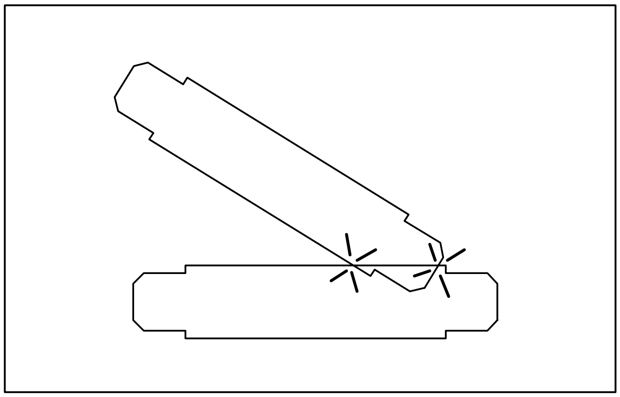

Figure 4 A collision between two AGVs is detected by an overlap of the two polygons describing the AGV shapes

To prove that two AGV polygons, each consisting of 16 segments, do not intersect, 16×16=256 combinations of two segments must be considered. Furthermore, if the total number of AGVs in the system is n, ½·n·(n-1) combinations of two AGVs must be checked for collision. In a typical terminal simulation, the total number of AGVs is 50. That means 1225 collision checks and 1225·256=313600 calculations for intersection of polygon segments!

Since a high overall simulation speed must be preserved, extra functionality is added to the collision check algorithm to let it operate more efficiently. One of the methods is adding a so-called enclosing circle to the AGV polygon. This is the smallest circle that encloses all the segments of the polygon. The use of these enclosing circles is as simple as it is effective: If two enclosing circles overlap (distance between circle centers < sum of radii), collision is possible and a precise collision check must be performed as described above. If the enclosing circles do not overlap, overlap of the polygons –and thus collision– is impossible.

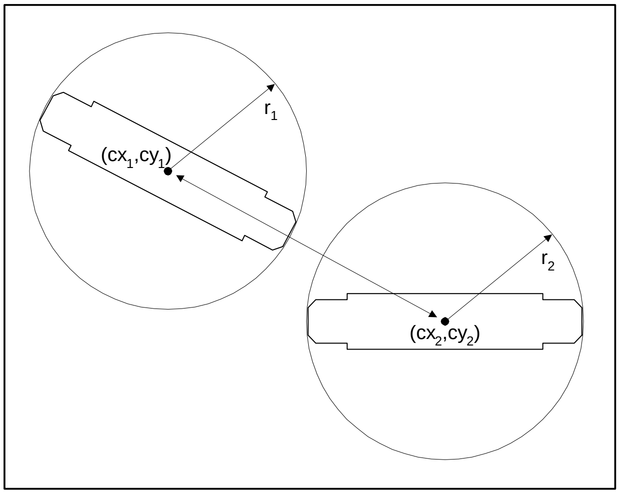

Figure 5 Adding enclosing circles around the AGV polygons to speed up the collision check

The frequency for activating the collision detection and distance monitoring algorithms depends on the maximum speed of the driving AGVs. For distance monitoring, the minimum time between checks is the time in which an AGV can drive half of the defined safety distance.

IMPLEMENTATION

TRACES is build as a prototype application. The simulation model is written in Pascal, and uses the extension MUST for the discrete event simulation. The software is compiled using the Borland Pascal compiler and runs on a Windows/Intel platform. Because optimizing the heuristics requires a lot of simulation runs speed is essential. A Pentium 200 MHz can complete about 30 different scenarios of average complexity (namely the current DSL-terminal at ECT Rotterdam) within 24 hours.

TRACES enables the user to quickly define a layout and routing with the use of nodes and connections. The routing should at least define one route between each quaycrane and stacking crane, and vice versa. Semaphores must be defined in critical zones to prevent collisions.

The terminal simulation package contains:

- models for the physical equipment : AGVs, Quay Cranes and Stacking Cranes
- a prototype of TRACES for traffic control
- a generator model for shiploads : containers to load and unload
- a control model for the planning, scheduling and dispatching of jobs
- a stack model, including algorithms for optimal stacking.

TRAVIS is developed on the same platform in Pascal/MUST. The first prototype uses output of the terminal simulation package, written to file. Simulation speed depends on the settings, mainly the size of the time interval for collision detection and the length of the safety distance. The module generates output for a 3D animation module.

The 3D animation module is written in Delphi, with use of the discrete event simulation tool TOMAS [Veeke]. Currently, the terminal simulation package is translated and redesigned in Delphi/TOMAS. This version will integrate TRAVIS and 3D animation in the package.

EFFECTS ON ENGINEERING EDUCATION

Students at the Delft University at the transportation department will get education in both the technical, physical aspects of transport and equipment, and in logistic control. In our automated container terminal research we find that these two aspects are highly correlated. The work on TRACES and TRAVIS is part of the lectures at the Delft University of Technology in the education for Mechanical Engineers at the Logistic Engineering Department. The lectures give students insight in the possibilities and difficulties of designing and controlling complex multi AGV systems and the application of automated systems at a container terminal.

Furthermore, students uses the technology in practical exercises. In one case, this has led to an industrial implementation of a TRACES based traffic control system for AGVs in a production environment.

The project resulted in a realistic 3D animation of the simulation model which can be used for both verification and validation, and communication with managers and operational personnel of an automated terminal.

FUTURE DEVELOPMENTS

Currently, the developed simulation models are redesigned and programmed in a Windows environment, using Delphi and the simulation package TOMAS. The result will be a distributed network of cooperating simulation models for quays, stacks, AGV system and shiploads, and in the future models for other modalities like barge, rail and truck service centers. The TRAVIS visualization will be an important part of that virtual world.

Students can build their own models to cooperate in this environment.

The department is also working on a laboratory of 1:25 scaled AGVs. In this laboratory, all aspects of terminal design can be experimented:

- AGV mechatronics
- wireless communication protocols
- traffic control
- terminal layout and routing
- job scheduling and dispatching
- high level terminal control

Our purpose is to control this AGV laboratory with our terminal simulation package.

CONCLUSIONS

The development of TRAVIS was done at the Delft University of Technology for research and educational purposes. The resulting software will give engineers and students insight in the possibilities and difficulties of designing and controlling complex multi AGV systems. The module gives instant feedback if operational errors, due to design errors, occur. Furthermore, the project resulted in a realistic 3D animation of the simulation model which can be used for both verification and validation, and communication with managers and operational personnel of an automated terminal. In the future, the concepts can be tested in a realistic, 1:25 scale, AGV laboratory.

REFERENCES

Ben-Ari, M. 1990. *Principles of Concurrent and Distributed Programming.* Prentice Hall.

Dijkstra, E.W. 1968. *Co-operating sequential processes.* In F. Genuys (ed.) Programming Languages, Academic Press, New York.

Duinkerken, M.B.; J.J.M. Evers; J.A. Ottjes. 1999. *TRACES : Traffic Control Engineering System.* Proceedings 31st Summer Computer Simulation Conference. Chicago [SCS] pp 461-465.

Duinkerken, M.B.; J.A. Ottjes. 2000. *A simulation model for automated container terminals.* Proceedings of the Business and Industry Simulation Symposium. Washington D.C. [SCS] pp 134-139.

Evers, J.J.M.; S. Koppers. 1996. *Automated guided vehicle traffic control at a container terminal.* Transportation Research (A). Vol. 30, no. 1, pp 21-34.

Evers, J.J.M.; G.J.M. van der Wielen, J.A. Ottjes, M.B. Duinkerken. 1998. *The jumbo container terminal: quay transport during ship loading*. TRAIL Studies 98/308, TRAIL Research School.

Terstegge, M.J.. 1999. *Control and collision detection for multi-AGV systems* (Master thesis; in Dutch). Delft University of Technology. report no. 99.3.LT.5191.

Veeke, H.P.M.; J.A. Ottjes. 2000. *TOMAS: Tool for Object-Oriented Modelling and Simulation.* Proceedings of the Business and Industry Simulation Symposium. Washington D.C. [SCS] pp 76-81.