Predicting the Impact Point of a Falling Body

Problem presented by

Stephen Cull

QinetiQ, Bedford

Problem statement

QinetiQ's Real Time All Vehicle Simulator (RTAVS) is a multi-platform simulation environment used for a variety of applications, including the simulation of a military fast jet. In this application, a Head Up Display (HUD) provides the pilot with navigation and weapons system information. The trajectory of any air to ground munition must be calculated many times a second in order to update the HUD information. Within RTAVS this calculation is performed along with several thousand others on a personal computer running the aircraft model and visuals in real-time. Therefore an efficient algorithm for the calculation of the predicted impact point is required, which must also be simple to implement, because the RTAVS software is maintained by several people all with differing mathematical knowledge.

Study Group contributors

David Burton (University of Lancaster)
Angela Mihai (University of Durham)
John Ockendon (University of Oxford)
Colin Please (University of Southampton)
Eddie Wilson (University of Bristol)
David Wood (University of Warwick)

Report prepared by

David Burton (d.burton@lancs.ac.uk)
Angela Mihai (l.a.mihai@durham.ac.uk)

1 Introduction

QinetiQ's Real Time All Vehicle Simulator (RTAVS) is a multi-platform simulation environment used for a variety of applications. A frequently used application is the simulation of a military fast jet. In this application a Head Up Display (HUD) provides the pilot with navigation and weapons system information. The weapons system information on the HUD gives an indication of the fall line and predicted impact point of any air to ground munition loaded. To facilitate this, the trajectory of the munition must be calculated many times a second in order to update the HUD information. On the real aircraft such a calculation would be performed by a dedicated unit. However, within RTAVS this calculation is performed along with several thousand others on a personal computer running the aircraft model and visuals in real-time. Therefore an efficient algorithm for the calculation of the predicted impact point is required. In addition to the efficiency issue, the algorithm also has to be simple to implement. This is because the RTAVS software is maintained by several people all with differing mathematical knowledge.

2 Physical model used in RTAVS

The physical model presented below was extracted from the source code that QinetiQ made available to the Study Group.

2.1 Equation of motion

Let $\mathbf{r} : \mathbb{R} \to \mathbb{R}^3$ be a position map where $p = \mathbf{r}(t)$ is a point representing the instantaneous position of the projectile, with the origin defined to be the release point of the projectile. The equation of motion for the projectile is[1]

$$m\ddot{\mathbf{r}}(t) = -\tfrac{1}{2}C_D(\dot{\mathbf{r}}(t), \mathbf{r}(t))\,\rho(\mathbf{r}(t))\,A|\dot{\mathbf{r}}(t)|\dot{\mathbf{r}}(t) + m\mathbf{g}\,, \tag{1}$$

where m is the mass of the projectile, $\mathbf{g}$ is the acceleration vector due to gravity, A is the cross-sectional area of the projectile across the surface with normal along $\dot{\mathbf{r}}(t)$, $\rho(\mathbf{p})$ is the air density at position $\mathbf{p}$ and $C_D(\dot{\mathbf{r}}(t), \mathbf{r}(t))$ is the instantaneous drag coefficient of the projectile. As usual, dots indicate differentiation with respect to time.

2.1.1 Air density and temperature

The air density is given as a function of the local air temperature T by

$$\tilde{\rho}(T) = \rho_0 \left(\frac{T}{T_0}\right)^{4.26}, \tag{2}$$

where $\rho_0 = 1.21\,\mathrm{kg\,m^{-3}}$ and $T_0 = 288.15\,\mathrm{K}$. In turn, T is given in terms of the altitude h of the projectile by

$$T(h) = T_0 - \frac{(6.5 \times 10^{-3})h}{1 + (1.6 \times 10^{-8})h}. \tag{3}$$

[1] Unless otherwise stated, all physical quantities in this article are in MKS units.

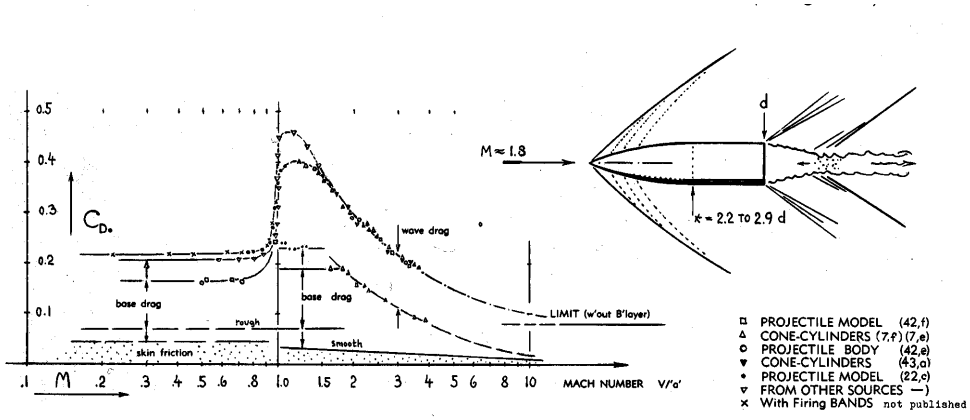

Figure 1: Typical values of the drag coefficient C_D versus the Mach number M for a projectile. This figure has been reproduced from reference [4].

2.1.2 Drag coefficient

The drag coefficient C_D is given as a function of the local Mach number M by

$$M(v,T) = \frac{v}{c_s(T)}, \tag{4}$$

$$c_s(T) = \sqrt{\frac{P(T)}{\tilde{\rho}(T)}}, \tag{5}$$

$$P(T) = 10^5 \left(\frac{T}{T_0}\right)^{5.26} \tag{6}$$

where c_s is the local speed of sound, v is the speed of the projectile and P is the local air pressure. Figure 1 indicates the behaviour of C_D over the subsonic to supersonic regimes. The important feature to note is the sharp increase in the drag coefficient immediately before Mach 1 is attained. Thus, there could be implications for the numerical schemes used to integrate (1) if the local sound barrier is crossed.

2.2 Typical values for the projectile properties

Typical values of constants in the problem are

$$A = 0.4\,\mathrm{m}^2, \tag{7}$$

$$m = 456\,\mathrm{kg} \tag{8}$$

with a typical release altitude of $10^4\,\mathrm{m}$.

2.3 Non-dimensional equation of motion

Let us non-dimensionalize (1) with respect to the release altitude l and the time scale $\tau = \sqrt{l/g}$. We obtain

$$\ddot{\hat{\mathbf{r}}}(\hat{t}) = -\kappa(\hat{h},\hat{v},l)|\dot{\hat{\mathbf{r}}}(\hat{t})|\dot{\hat{\mathbf{r}}}(\hat{t}) + \mathbf{k}\,. \tag{9}$$

Here $\mathbf{k}$ is the downward unit vector and $\hat{X}$ indicates the dimensionless quantity X/x, where $x = l^a\tau^b$ for some $a, b \in \mathbb{R}$. The level of influence of the aerodynamic force on the projectile is dictated by the dimensionless function

$$\kappa(\hat{h}, \hat{v}, l) = \frac{\tilde{\rho} \circ T(l\hat{h})A}{2m} C_D \circ M(l\hat{v}/\tau, T(l\hat{h}))l \tag{10}$$

which is positive-definite and turns out to be $O(1)$ for our regime of interest.

3 Methodology

Our proposed solution involves

- estimating a suitable time step based on the initial conditions and the exact free-fall solution to (9) for the case of constant κ,
- using a look-up table to find $\kappa(\hat{h}, \hat{v}, l)\hat{v}$ and
- using an integration scheme that is a trade-off between run-time efficiency and ease of implementation.

3.1 A guess for the time step via the free-fall solution

An order of magnitude estimate for the predicted impact time is obtained by considering the initial conditions $\mathbf{r}(0) = \dot{\mathbf{r}}(0) = (0\ 0\ 0)$ with κ set to a constant. Then (9) collapses to the single component equation

$$\ddot{Z}(\hat{t}) = -\kappa \dot{Z}(\hat{t})^2 + 1 \tag{11}$$

for the dimensionless distance $Z = \hat{\mathbf{r}} \cdot \mathbf{k}$ of the projectile from its point of release. The solution corresponding to the initial conditions $Z(0) = 0$ and $\dot{Z}(0) = 0$ is

$$Z(\hat{t}) = \frac{1}{\kappa} \ln\big[\cosh(\sqrt{\kappa}\hat{t})\big]. \tag{12}$$

The time to impact, τ_0, is obtained by setting $Z(\tau_0) = 1$ in (12) and rearranging this expression to give

$$\tau_0 = \frac{1}{\sqrt{\kappa}} \cosh^{-1}\big[\exp(\kappa)\big]. \tag{13}$$

One now divides τ_0 by a number, depending on the integration scheme, to give a suitable time step.

3.2 Look-up table

In order to avoid unnecessary computation the look-up table for $\kappa(\hat{h}, \hat{v}, l)\hat{v}$ should be specified in terms of v^2 rather than $\hat{v}$. For ease of maintainence the computer code should be implemented using the dimensionful equation of motion, and, wherever possible, square roots should be avoided because of their slow computation time.

3.3 Integration scheme

Again, when selecting an integration scheme not only must we bear in mind its run-time efficiency but also that it should be as easy as possible to implement and maintain by a variety of programming teams. Possible schemes include Runge-Kutta (with an adaptive step size [6] to cope with crossing the local sound barrier) or the Galerkin method. An implementation of the latter is the focus of the rest of this article.

4 Numerical integration of the equation of motion

For notational simplicity let us drop the *hats* from all dimensionless quantities. Equation (9), with suitable initial conditions, then reads

$$\begin{aligned} \ddot{\mathbf{r}}(t) &= -\kappa|\dot{\mathbf{r}}(t)|\,\dot{\mathbf{r}}(t) + \mathbf{k},\ t \in [0, \tau_0] \\ \mathbf{r}(0) &= (0\ 0\ 0),\ \dot{\mathbf{r}}(0) = (V\ 0\ 0)\,, \end{aligned} \tag{14}$$

where V is the initial speed of the projectile. We implemented a *second-order Runge-Kutta* method [6] and the results were found to be quite satisfactory when no drastic sudden changes occurred in the value of $\kappa|\dot{r}(t)|$. The *second-order initial value problem* (14) was reduced to two coupled *first-order equations* of the form

$$\begin{aligned} \dot{\mathbf{r}}(t) &= \mathbf{R}(t), \\ \dot{\mathbf{R}}(t) &= -\kappa|\mathbf{R}(t)|\,\mathbf{R}(t) + \mathbf{k}. \end{aligned} \tag{15}$$

Note that an *embedded Runge-Kutta* [6] approach can be employed for the adaptive step-size control near and across the sound barrier. In this report we propose a *finite element* method for solving equation (14). This has the advantage that it can be applied to a more general class of problems with rapidly changing solutions. Furthermore, the algorithm is straightforward to implement.

Let the estimated time for the object to reach the ground, τ_0, be *the unit for time*. We also consider the second component, Y, of the vector $\mathbf{r}(t) = (X(t)\ Y(t)\ Z(t))$ to be constant and equal to 0, and therefore reduce (14) to a *two-component equation* of the form

$$\begin{aligned} \ddot{r}(t) &= -\kappa|\dot{r}(t)|\,\dot{r}(t) + k,\ t \in [0, 1] \\ r(0) &= (0\ 0),\ \dot{r}(0) = (V\ 0), \end{aligned} \tag{16}$$

where $r(t) \equiv (X(t)\ Z(t))$ and $k \equiv (0\ 1)$. For computational speed, the coefficient $\kappa|\dot{r}(t)|$ is tabulated as a function of the height and velocity of the projectile. Problem (16) has *two initial conditions*: an homogeneous Dirichlet condition, $r(0) = (0\ 0)$, given by the position of the aircraft at the moment when the projectile is released, and a Neumann condition, $\dot{r}(0) = (V\ 0)$, which is given by the velocity of the aircraft at the moment of release. We also know the vertical component of the trajectory described by the falling object at the end of the time interval is $Z(1) = l$. Here l is the altitude of the airplane at the moment when the object is released.

4.1 The discretization of the problem

First we consider a partition of the time interval $[0,1]$ into N ($N > 1$) disjoint elements. For the algorithm considered in this report the size of the elements can be quite arbitrary. However, to simplify the presentation, we restrict our attention to the case where the mesh-size, h, is uniform and equal to $\frac{1}{N}$.

The *discrete form of equation* (16) corresponding to the N finite elements is

$$\begin{aligned} \ddot{r}_N(t) + \kappa_N |\dot{r}_N(t)| \dot{r}_N(t) &= (0\ 1),\ t \in [0,1] \\ r_N(0) = (0\ 0),\ \dot{r}_N(0) &= (V\ 0). \end{aligned} \tag{17}$$

It is important to appreciate that $r_N(t) = r(t)$ *only* if $t \in (1/N, 2/N, \ldots, 1)$, and that $r_N(t) \to r(t)$ in the limit $N \to \infty$. Let r_0 be an initial guess and the following recurrence formula hold for a sequence of approximations (r_n) to the discrete solution r_N:

$$\begin{aligned} \ddot{r}_{n+1}(t) + \kappa_n |\dot{r}_n(t)|\, \dot{r}_{n+1}(t) &= (0\ 1),\ t \in [0,1] \\ r_{n+1}(0) = (0\ 0),\ \dot{r}_{n+1}(0) &= (V\ 0). \end{aligned} \tag{18}$$

This is equivalent to

$$\begin{aligned} \ddot{r}_{n+1}(t) + b_n \dot{r}_{n+1}(t) &= (0\ 1),\ t \in [0,1] \\ r_{n+1}(0) = (0\ 0),\ \dot{r}_{n+1}(0) &= (V\ 0), \end{aligned} \tag{19}$$

where $b_n = \kappa_n |\dot{r}_n|$.

In our approach, we first change equation (19) into a *two-boundary problem*, then *adjust the boundaries* so that the initial conditions are satisfied. The discrete problem to solve is now

$$\begin{aligned} \ddot{r}_{n+1}(t) + b_n \dot{r}_{n+1}(t) &= (0\ 1),\ t \in [0,1] \\ r_{n+1}(0) = (0\ 0),\ r_{n+1}(1) &= (\alpha\ l), \end{aligned} \tag{20}$$

where $(\alpha\ l)$ is a guess for the final position of the projectile when it hits the ground. Note that we only have to make a guess for the horizontal coordinate α since the vertical coordinate l is known. Once we find the solution to (20) we then adjust it so that $\dot{r}(0) = (V\ 0)$.

4.2 A Petrov-Galerkin finite element approach

Upon dropping the subscripts in (20), the variational component-problems read: for the X-component, find $X \in H^1([0,1])$ such that

$$\begin{aligned} L(X, w) = 0,\ \forall w \in H_0^1([0,1]), \\ X(0) = 0,\ X(1) = \alpha, \end{aligned} \tag{21}$$

and, for the Z-component, find $Z \in H^1([0,1])$ such that

$$\begin{aligned} L(Z, w) = (1, w),\ \forall w \in H_0^1([0,1]), \\ Z(0) = 0,\ Z(1) = l, \end{aligned} \tag{22}$$

where

$$L(u,w) = -(\dot{u},\dot{w}) + b\cdot(\dot{u},w)$$

and

$$(u,w) = \int_0^1 u\cdot w dt$$

for all $u, v \in H^1([0,1])$. Here $H^1([0,1])$ denotes the usual Sobolev space defined by the seminorm

$$|u|^2 = \int_0^1 \dot{u}\dot{u}dt$$

and the norm

$$\|u\|^2 = |u|^2 + \|u\|_{L^2([0,1])}.$$

In order to solve equations (21) and (22) we employ the *exponential upwinding Petrov-Galerkin method*.

The Petrov-Galerkin method consists of taking two finite-dimensional subspaces $V, W \subset H^1([0,1])$ (known as *the trial space* and the *the test space*, respectively), where $\dim(V) = \dim(W)$, and solving *the discrete weak form*: find $u \in V$ such that

$$L(u,w) = (f,w),\ \forall w \in W.$$

Let V be the space of continuous piecewise linear functions generated by the basis functions

$$\varphi_i(t) = \begin{cases} \frac{h+t-t_i}{h} & \text{if } t \in [t_{i-1}, t_i] \\ \frac{h-t+t_i}{h} & \text{if } t \in [t_i, t_{i+1}] \\ 0 & \text{if } |t-t_i| > h \end{cases}$$

where $t_i = \frac{i}{N}$ $(i = 0, \cdots, N)$ are nodes in an uniform time-mesh and $h = \frac{1}{N}$ is the mesh-size.

A new *test space* W is introduced in [5], which is defined as $W = \text{span}\{\psi_i\}$, where $i = 0, \cdots, N$, and for each ψ_i the following properties hold:

(1) ψ_i is continuous in $[0,1]$,

(2) $\psi_i = 1$ at node t_i,

(3) $\psi_i = 0$ in all elements for which t_i is not a vertex,

(4) $\ddot{\psi}_i - b\dot{\psi}_i = 0$ within each element having node t_i as a vertex.

Hence W is the space of continuous functions generated by *the piecewise exponential basis functions*

$$\psi_i(t) = \begin{cases} \frac{1-e^{b(h+t-t_i)}}{1-e^{bh}} & \text{if } t \in [t_{i-1}, t_i] \\ \frac{e^{b(t-t_i)}-e^{bh}}{1-e^{bh}} & \text{if } t \in [t_i, t_{i+1}] \\ 0 & \text{if } |t-t_i| > h \end{cases}$$

for all interior nodes, t_i, of the uniform mesh with mesh-size h.

Let $x_i := X(t_i)$ for all $i = 0, \cdots, N$. We can expand X in terms of the nodal basis functions φ_i:

$$X = x_0 \cdot \varphi_0 + \sum_{i=1}^{N-1} x_i \cdot \varphi_i + x_N \cdot \varphi_N.$$

From the assumed boundary conditions we deduce

$$\begin{aligned} X(0) &= x_0 = 0, \\ X(N) &= x_N = \alpha. \end{aligned}$$

We then choose w to be each of the basis functions $\psi_1, \cdots, \psi_N$ in turn, which we substitute into (21).

The variational problem (21) can now be written as a sparse linear system:

$$M\underline{x} = 0, \tag{S_1}$$

where $\underline{x} = (x_j)_{j=0,\cdots,N}^T$ and M is a tridiagonal matrix because the supports of the nodal basis functions overlap only for the nearby nodes. Explicitly $M = (m_{ij})$, where

$$m_{ij} = -(\dot{\varphi}_i, \dot{\psi}_j) + b \cdot (\dot{\varphi}_i, \psi_j), \ i, j = 0, \cdots, N\,.$$

With the guessed value $x_N = \alpha$, the system (S_1) becomes

$$\begin{aligned} m_{11}x_1 + m_{12}x_2 &= 0 \\ m_{21}x_1 + m_{22}x_2 + m_{23}x_3 &= 0 \\ &\cdots \\ m_{N-1,N-2}x_{N-2} + m_{N-1,N-1}x_{N-1} &= -m_{N-1,N}\alpha, \end{aligned} \tag{S_1'}$$

which can be solved by, for example, Gaussian elimination.

Now, the initial conditions for the equation of motion must be satisfied. Therefore we consider

$$\dot{X}(0) = -x_0 \cdot \frac{1}{h} + x_1 \cdot \frac{1}{h} = V.$$

If $x_1 = V \cdot h$, then the problem (21) is solved. However, if $x_1 \neq V \cdot h$ then the vector solution to the system (S_1'), $(x_1 \ x_2 \ \cdots \ x_N)$, must be multiplied by $\frac{V \cdot h}{x_1}$ for the next iteration.

For the Z-component, if $z_i := Z(t_i)$ for all $i = 0, \cdots, N$, then Z can be expanded in terms of the nodal basis functions φ_i as follows:

$$Z = z_0 \cdot \varphi_0 + z_1 \cdot \varphi_1 + \sum_{i=2}^{N-1} z_i \cdot \varphi_i + z_N \cdot \varphi_N.$$

The boundary conditions are

$$\begin{aligned} Z(0) &= z_0 = 0 \\ Z(N) &= z_N = l. \end{aligned}$$

We then choose w to be each one of the basis functions $\psi_2, \cdots, \psi_N$ in turn, which we substitute into (22).

The variational problem (22) reduces to a sparse linear system:

$$M\underline{z} = \underline{f}, \tag{S_2}$$

where $\underline{z} = (z_j)_{j=0,\cdots N}^T$, M is the same as in (S_1) and

$$\underline{f} = (f_j)_{j=0,\cdots,N}^T,\ f_j = (1, \psi_j),\ j = 0, \cdots, N\,.$$

The second initial condition gives us

$$\dot{Z}(0) = -z_0 \cdot \frac{1}{h} + z_1 \cdot \frac{1}{h} = 0.$$

In practice, we can replace $z_1 = 0$ by $z_1 = \mathcal{O}(h^2)$ if we wish.

Since $z_N = l$, the system (S_2) becomes

$$\begin{gathered} m_{22}z_2 + m_{23}z_3 = f_2 \\ \cdots \\ m_{N-1,N-2}z_{N-2} + m_{N-1,N-1}z_{N-1} = f_{N-1} - m_{N-1,N}l. \end{gathered} \tag{S_2'}$$

Proposed Algorithm. Let N be the number of elements in the time-mesh and let $h = 1/N$. Given $(X_n\ Z_n)$, an approximation to the solution $(X_N\ Z_N)$ of the discrete equation (17), we define the next iterate $(X_{n+1}\ Z_{n+1})$ as follows, where zeros(n, m) (respectively ones(n, m)) indicates an $n \times m$ matrix with every entry equal to 0 (respectively 1).

(1) Set $(u_0^x\ u_0^z) = (X_n\ Z_n)$;

(2) Set $M = \text{zeros}(N + 1, N + 1)$;
For $j = 0$ until convergence
For $i = 1 : N$

$$m = \begin{pmatrix} -\dfrac{b \cdot e^{bh}}{1 - e^{bh}} & \dfrac{b \cdot e^{bh}}{1 - e^{bh}} \\ \dfrac{b}{1 - e^{bh}} & -\dfrac{b}{1 - e^{bh}} \end{pmatrix}$$

$$M(i : i + 1, i : i + 1) = M(i : i + 1, i : i + 1) + m;$$

End i
Set the right-hand side of the equation:

$$\begin{aligned} \underline{f}_x &= \text{zeros}(N - 1, 1) \\ \underline{f}_z &= \text{ones}(N - 2, 1), \end{aligned}$$

The boundary conditions give that

$$\begin{aligned} f_x(N - 1) &= f(N - 1) - \alpha \cdot M(N - 1, N) \\ f_z(N - 2) &= f(N - 2) - l \cdot M(N - 1, N). \end{aligned}$$

Calculate the solutions to the sparse linear systems

$$\underline{v}^x = M(2:N, 2:N)^{-1} \cdot \underline{f}_x$$
$$\underline{v}^z = M(3:N, 3:N)^{-1} \cdot \underline{f}_z;$$

if $v^x(1) \neq V \cdot h$ then

$$\underline{v}^x = \underline{v}^x \cdot \frac{V \cdot h}{v^x(1)};\ \alpha = \alpha \cdot \frac{V \cdot h}{v^x(1)}$$

End if.
Set $u^x_{j+1} = (0\ \underline{v}^x\ \alpha)$; $u^z_{j+1} = (0\ 0\ \underline{v}^z\ l)$.
End j.

(3) $(X_{n+1}\ Z_{n+1}) = (u^x_{j+1}\ u^z_{j+1})$. □

References

[1] Brenner, S. C. and Scott, L. R. *The Mathematical Theory of Finite Element Methods.* Springer-Verlag, New York, 1994.

[2] Eriksson, K., Estep, D., Hansbo, P. and Johnson, C. *Computational Differential Equations.* Cambridge University Press, 1996.

[3] Golub, G. H. and van Loan, C. F. *Matrix Computations.* Johns Hopkins, Baltimore, 1989.

[4] Hoerner, S. F. *Fluid-Dynamic Drag.* Published by the Author, 1965.

[5] Perella, A. J. *A Class of Petrov-Galerkin Finite Element Methods for The Numerical Solution of The Stationary Convection-Diffusion Equation.* Ph.D. thesis, University of Durham, UK, 1996.

[6] Press, W. H., Teukolsky, S. A., Vetterling and W. T., Flannery, B. P. *Numerical Recipes in C, 2nd ed.* Cambridge University Press, 1995.

[7] Strang, G. and Fix, G. *An Analysis of the Finite Element Method.* Prentice-Hall, Englewood Cliffs, NJ, 1973.