# 10 Fingers of Death: Algorithms for Combat Killing

*Roger Smith and Don Stoner – Titan Corporation*

Good shooting games need good killing algorithms. This gem provides a series of combat algorithms that can be used to improve the realism of combat decisions and do so with faster algorithms. Most of the algorithms discussed here were developed for the United States military and have been validated for use in one or more real combat simulations.

First-person shooter killing algorithms are fine, but some situations can be handled more accurately and efficiently by including geometry, statistics, probability, and aggregation. Massively Multi-player and Real-time Strategy games particularly include a lot of action that does not have to be modeled with individual line-of-sight (LOS) and targeting for a headshot. These and other games can also benefit from the inclusion of multiple kill types that are based on real live-fire experiments. First-person shooters may equip AI's with some of these algorithms while leaving the more detailed LOS algorithms for the avatar controlled by the human player.

## Hitting a Ribbon

The first finger of death presents a simple method for determining whether a shooter will hit a ribbon target like a road, river, long convoy of vehicles, or a serpentine creature (Figure 1). If the target is so long that it is impossible to overshoot or undershoot its length, the probability of hitting the target is dependent only upon the width of the target and the standard deviation of the shot pattern of the weapon. This simple algorithm is also a good way to introduce the logic and mathematics behind several of the attrition algorithms that follow [Parry95]. Deviations in the impact point of the munitions being fired are due to factors such as the quality of the weapon, steadiness and skill of the human operator, variations in the construction of the projectile, and wind conditions.

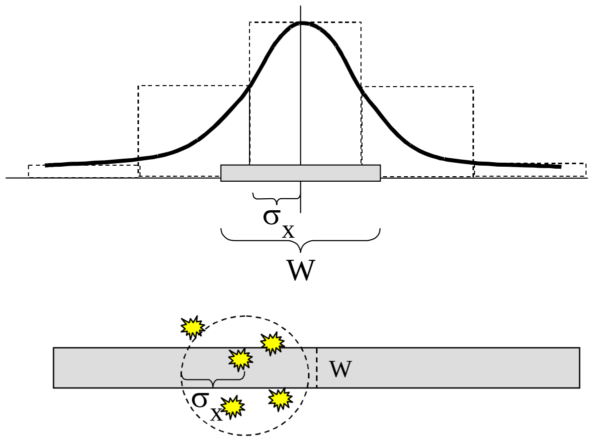

Figure 1. Probability of Hitting a Ribbon Target

### *Equation*

The equation for calculating the probability of hit ($P_h$) for a ribbon target is:

$$P_h = 2W \Big/ \sqrt{2\pi\sigma_x}$$

where,

W is the width of the target, and
$\sigma_x$ is the standard deviation of the bullet dispersion in the x dimension.
This assumes that the pattern is normally distributed with the same standard deviation in both the x and y dimensions.

[Note: The code for all of these algorithms can be found on the CD-ROM.]

## Hitting the Bullseye

The second finger of death describes the math and probability of hitting a round target. Like the previous algorithm, this one is based on the fact that all shooters, human and machine alike, have built-in variation in every shot fired.

The algorithm is driven by two very simple variables – the radius of the target and the standard deviation of the rounds. This deviation is based on a normal distribution in which the mean value is zero because the shooter is aimed directly at the center of the target [Parry95]. The algorithm determines whether each shot will hit the target, but does not calculate the actual impact point of the round. This simplification eliminates calculations that would have to be done to distribute the round normally in both the x and y dimension.

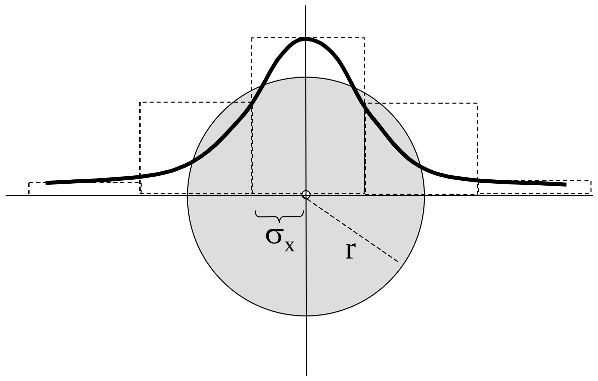

Figure 2. Probability of Hitting the Bullseye

### *Equation*

The equation for calculating the probability of hit ($P_h$) for a round target is:

$$P_h = 1 - e^{-\left(r^2/2\sigma_x^{\ 2}\right)}$$

where,

r is the radius of the target, and

$\sigma_x$ is the standard deviation of the bullet dispersion in the x dimension.

## Hitting a Rectangle

Most targets are not shaped like bullseyes, so we need a more flexible algorithm to shoot rectangular targets like the torso of a human or a vehicle. This algorithm includes measures for the length and width of a rectangular target [Parry95].

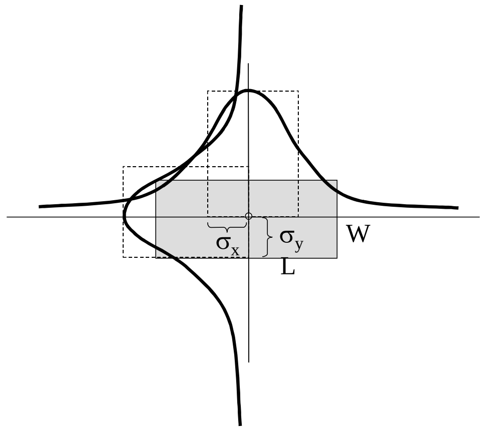

Figure 3. Probability of Hitting a Rectangle

### *Equation*

The equation for calculating the probability of hit ($P_h$) for a rectangular target is:

$$P_h = \sqrt{A * B}$$
$$A = 1 - e^{-\left(2L^2/\pi\sigma_x{}^2\right)}$$
$$B = 1 - e^{-\left(2W^2/\pi\sigma_y{}^2\right)}$$

where,

L is the length of the target in the x dimension,

W is the width of the target in the y dimension,

$\sigma_x$ is the standard deviation of the bullet dispersion in the x dimension, and

$\sigma_y$ is the standard deviation of the bullet dispersion in the y dimension.

Weapons often have different standard deviations in the x and y dimensions. For example, when a football quarterback throws a pass, the variation from the aim point along the axis of flight is usually greater than the variation left or right of the aim point. The same is true for missiles being fired at a combat vehicle or rocks being thrown at a dinosaur.

## Shotgunning a Small Target

Some weapons unleash a barrage of rockets, bomblets, or explosive munitions all at once in an attempt to totally overwhelm the target and blow it to smithereens. When this happens, there are much faster ways of determining the killing effect of the entire barrage than calculating the impact points and lethality of each rocket individually and then accumulating them.

This algorithm calculates the probability that one of the munitions' lethal areas will overlap with the point target. The size of the target is not considered in these calculations because it is assumed that the lethal blast area can encompass an entire target [Parry95].

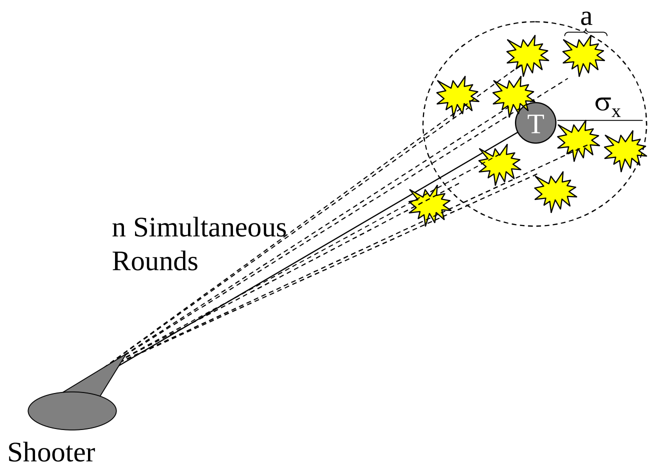

Figure 4. Probability of Killing a Target with a Simultaneous Barrage of Munitions

### *Equation*

$$P_k = 1 - e^{-\left(na/2\pi\sigma_x^{\;2}\right)}$$

where,

n is the number of rounds in the barrage,
a is the lethal area of a single round against this target,  and
$\sigma_x$ is the standard deviation of the bullet dispersion in the x dimension.

## Death by Walking Artillery

Artillery and catapult rounds are often adjusted by a spotting team that radios corrections back to the firing battery and allows them to place the next round closer to the target. When this occurs, the lethality of the barrage is higher than the previous shotgunning method. The lethality of this walking artillery is calculated through a summation series in the exponent [Parry95].

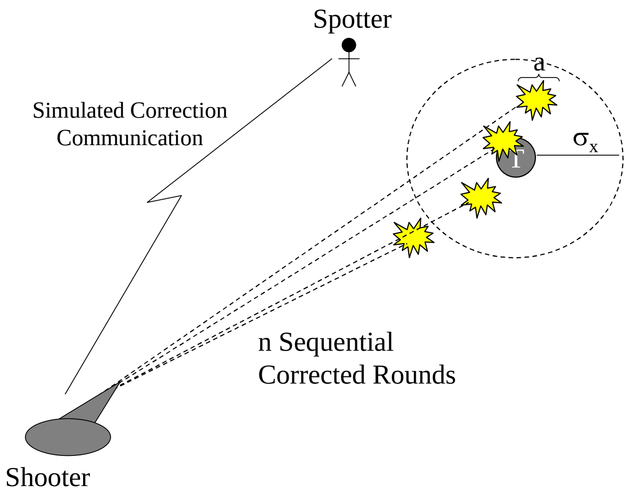
Spotter
a
$\sigma_x$
T
Simulated Correction Communication
n Sequential Corrected Rounds
Shooter

Figure 5. Lethality of Walking Artillery with a Spotter

*Equation*

$$P_k = 1 - e^{-\left(\left(a^2/2\sigma_x{}^2\right)*\sum_{i=2}^{n}(i-1)/i\right)}$$

where,

n is the number of rounds fired in the barrage,
a is the lethal area of a single round against this target,  and
$\sigma_x$ is the standard deviation of the bullet dispersion in the x dimension.

## Kills Come in Four Flavors

To paraphrase a famous pig, “all kills are equal, but some are more equal than others.” Military simulations usually model four different types of kills that are most often found in real-world combat. The first flavor is a mobility kill in which the target is no longer able to move, but remains alive enough to fire its weapon or communicate with other vehicles. The second is a firepower kill in which the weapon is damaged, but the vehicle or person is still able to move. The third is a mobility *and* firepower kill in which the vehicle or person is still alive, but cannot move or use its weapon. This target may still be able to observe enemy operations, communicate, consume supplies and, in some simulations, trigger a rescue operation. The final kill type is the catastrophic kill or K-kill, often pictured as an aircraft exploding into a million pieces, a flaming tank turret spinning through the air, or a person being turned into fresh chunks of meat.

These four kill types can be pictured as a Venn diagram (Figure 6). Though this form clearly communicates the relationships between the kill types, in order to be applied, it has to be separated so that a specific kill type can be determined quickly for each engagement. This separated data is usually represented as a kill thermometer (Figure 7). Normalizing the kill types in a single space as represented in the thermometer allows a program to determine the kill type of an engagement by drawing a single random number.

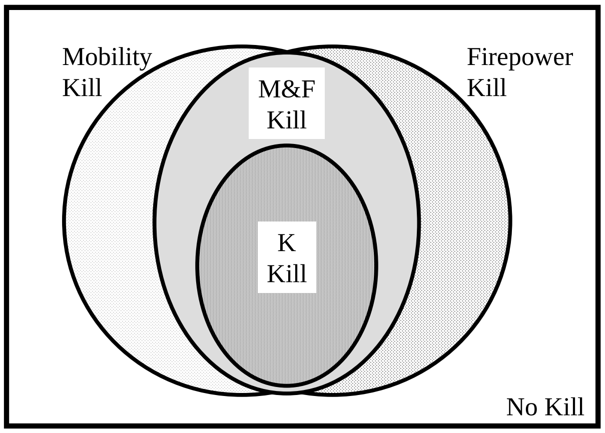

Figure 6. Standard Kill Types

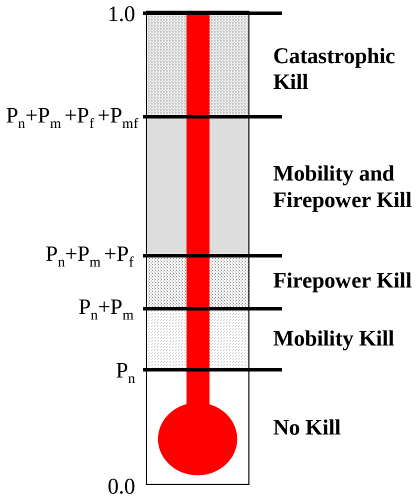

Figure 7. Kill Thermometer

There are live-fire projects that determine the probability of each kill type under different conditions by firing real weapons at real targets and measuring the result. Most simulations and games do not have access to such a rich source of information. Therefore, we have to identify common trends in experimental data and create equations that mimic those while remaining flexible enough to be applied to new weapon/target pairs. One simulation project noticed a distinct relationship between the mobility, firepower, and catastrophic kill data they had received from live-fire experiments. This relationship allowed them to create simple equations that use the root of a single "probability of mobility or firepower kill" ($P_{MoF}$ is the union of all of the shaded areas above) to calculate all of the other probabilities. The trend they noticed was that a mobility kill occurred 90% of the time that damage was done ($P_M = 0.9* P_{MoF}$); a firepower kill occurred 90% of the time ($P_F = 0.9* P_{MoF}$); and a catastrophic kill occurred 50% of the time that damage was done ($P_K = 0.5* P_{MoF}$).

However, this information cannot be applied directly to the kill thermometer in Figure 7. $P_M$ does not say that 90% of all engagements result in a mobility kill. It says that 90% of mobility-or-firepower kills include a mobility kill. Therefore, it has to be separated to make it possible to draw a single random number and determine which kill to apply to the target. These independent kill probabilities can be extracted as shown below.

### *Equation*

$$P_n = 1.0 - P_{MoF}$$
$$P_m = P_{MoF} - P_F = 0.1 * P_{MoF}$$
$$P_f = P_{MoF} - P_M = 0.1 * P_{MoF}$$
$$P_k = 0.5 * P_{MoF}$$
$$P_{mf} = P_{MoF} - P_m - P_f - P_k = 0.3 * P_{MoF}$$

where, the small subscript indicates the probability that only one type of kill occurs. For example, $P_m$ is the probability of *only* getting a mobility kill, but not getting any other form of kill. $P_n$ is the probability of no kill occurring.

These independent kill probabilities determine where the breakpoints fall in a kill thermometer and can be easily programmed as shown in the code on the CD-ROM.

## Chemicals, Fireballs, and Area Magic

There have been many models of the dispersion of chemicals and other agents. The following simple algorithm calculates the probability of a kill based on the volume of chemical released and the distance that the release occurs from the target. For games, this algorithm could be used for expanding fireballs, area magic, or any other exotic and evil weapon.

### *Equation*

$$P_k = \left(\sqrt[3]{nw_r} \big/ \sqrt{2\pi}\right) * e^{-0.5*\left(k*d^2/nw_r\right)^2}$$

where,

n is the number of rounds impacting at a specific point,
$w_r$ is the weight of the chemical inside of each round (in kilograms),
d is the distance that the rounds fall from the target location (in meters), and
k is a constant representing the dispersion characteristic of the chemical. For these experiments we recommend beginning with a value of 0.00135.

This equation allows you to deal with each round individually or to aggregate multiple rounds into a single attack centered at the same impact point. The equation also incorporates the constant k that represents the density and viscosity of a chemical compound. You can adjust this value to create the effect desired.

## The Shrapnel Wedge

When an aircraft is shot down with a missile it is seldom accomplished by the missile flying directly into the aircraft. More often, the missile reaches a “point of closest approach” and explodes near the aircraft. The shrapnel from the missile then spreads out in a donut or spherical pattern from the point of explosion and hopefully, the aircraft is caught in that shrapnel pattern and destroyed [Ball85]. This algorithm can be used with exploding projectiles, fireballs, and magic targeted at aircraft, dragons, and spacebugs.

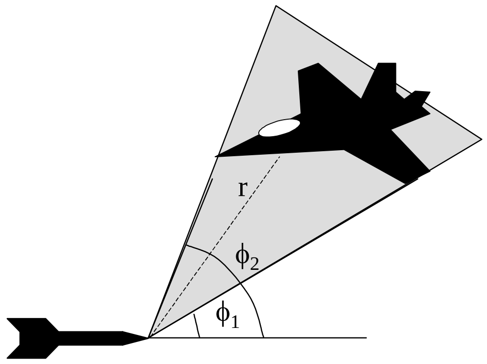

Figure 8. Probability of Killing a Target in a Shrapnel Wedge

### *Equation*

$$x = nA_v / \left(2\pi r^2 \left(\cos\phi_1 - \cos\phi_2\right)\right)$$
$$P_k = 1 - e^{-x}$$

where,

n is the number of fragments or projectiles in the missile warhead,
$A_v$ is the vulnerable area of the target presented to the missile (in sq meters),
r is the range from the detonation point to the target (in meters),
$\phi_1$ is the angle from the trajectory of the missile to the near edge of the vulnerability area of the target, and
$\phi_2$ is the angle from the trajectory of the missile to the far edge of the vulnerability area of the target.

## Beating the Bushes

Some engagements involve teams of hunters searching the terrain or bushes for hidden prey [Shubik83]. When a large group of hunters is looking for a large group of prey, it is possible to model the capture or kill of the prey in an aggregate form, rather than representing the individual movement and line-of-sight of every hunter and every prey.

As before, this approach is very valuable when the hunting and killing is being conducted by AI controlled hunters and especially when it is happening off the player's screen.

The algorithm is structured to calculate the change in the population of the prey based on the number and efficiency of the hunters. It also accounts for different types of prey and hunter animals, e.g. small rodents, medium-sized wolves, and large elephants.

To use the algorithm, we must define a probability of detection for each type of hunter against each type of prey under the given conditions (open terrain, forest, city, etc.). We must also select a "hardness" factor that differentiates the ability of the prey to elude, escape, or survive the actions of the hunter. These numbers are usually determined heuristically through experimentation and observation.

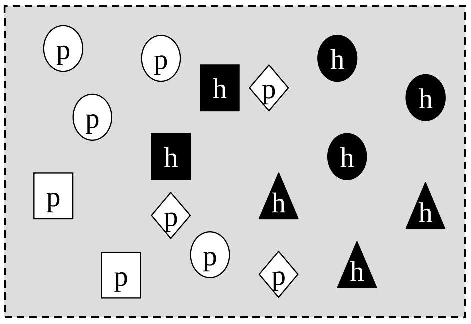

Figure 9. Multiple Types of Hunters Searching for Multiple Types of Prey

*Equation*

$$x = \left( \left( k_j / p \right) * \sum_{i=1}^{n} D_{i,j} * h_i \right)$$

$$A_j = p_j * (1 - e^{-x})$$

where,

$A_j$ is the number of kills of animal type j,
$p_j$ is the number of prey of type j,
$k_j$ is a hardness measure of the prey in the range [0,1],
p is the total number of prey of all types,
n is the number of prey types
$D_{i,j}$ is the probability that a hunter of type i can detect a prey of type j, and
$h_i$ is the number of hunters of type i.

## Beating the Bushes with Prey Spacing

The final finger of death is a modification of the previous one. Mathematicians and analysts noticed that the previous algorithm did not account for differences in the density of prey hiding in the bushes. It is clearly much easier to find and kill prey when there are a hundred of them in the search area than if there are just two or three. Therefore, they created a variation known as the Lulejian model [Shubik83] in which the spacing between the prey is an important factor. The visual picture for this algorithm is the same as that above, but the mathematics differ to account for the spacing of prey. The definition of $k_j$ also varies slightly in that Lilejian defines $k_j$ as the average destruction of the hunters on prey type j.

*Equation*

$$x = \left( \sum_{i=1}^{n} k_j * h_i \right) \Big/ (s * p)$$

$$A_j = p_j * (1 - e^{-x})$$

where,

$A_j$ is the number of kills of prey type j,
$p_j$ is the number of prey of type j,
s is the average spacing between the prey in the search area (in meters),
p is the total number of prey of all types,
n is the number of prey types
$k_j$ is the average destruction of the hunters on prey type j, in the range [0,1],
$h_i$ is the number of hunters of type i.

## Conclusion

The ten fingers of death described in this chapter are just a few of the combat killing algorithms that can be applied to computer games. The concepts of geometry, probability, statistics, and physics used in the ten fingers of death are good examples of approaches to many problems. Game developers should do what military modelers do to improve these – apply experience, mathematics, creativity, and other sciences to find equations that work well for your game. Don't be afraid to experiment!

## References

[Ball85] Ball, Robert E. *The Fundamentals of Aircraft Combat Survivability Analysis and Design*. AIAA Press, 1985.
[Epstein85] Epstein, Joshua M. *The Calculus of Conventional War: Dynamic Analysis without Lanchester Theory*. Brookings Institution, 1985.
[Parry95] Parry, Samuel, Editor. *Military OR Analyst's Handbook: Conventional Weapons Effects*. Military Operations Research Society, 1995.
[May02] May, Janet O. "OneSAF Killer/Victim Scoreboard Capability for C2 Experimentation", *Proceedings of the 2002 Conference on Behavioral Representation in Modeling and Simulation*, 2002.
[Shubik83] Shubik, Martin, Editor. *Mathematics of Conflict*. Elsevier Science Publishers, 1983.
[Army90] U.S. Army. *Field Artillery Handbook*. U.S. Army, 1990.