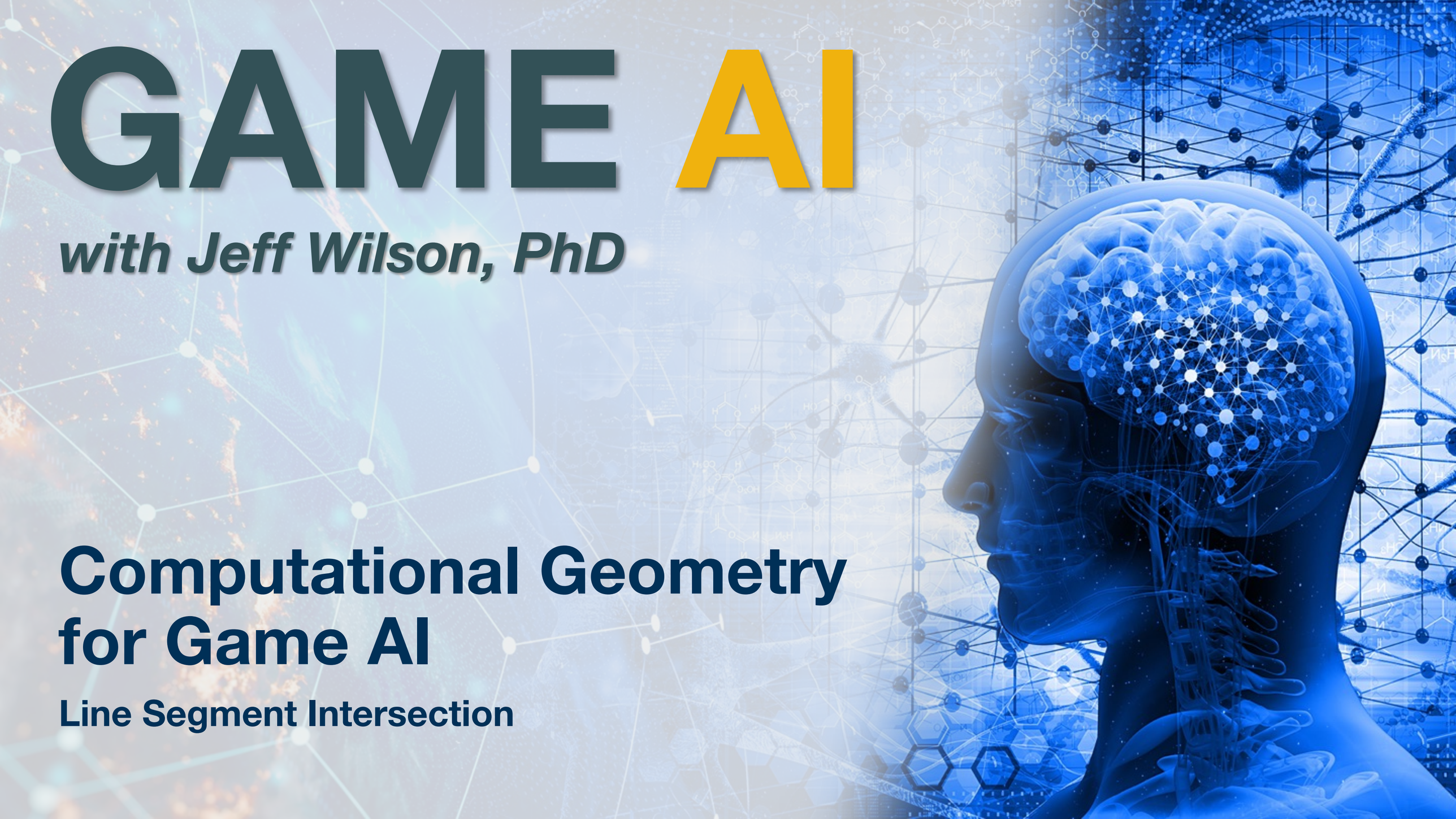
GAME AI

with Jeff Wilson, PhD



Computational Geometry for Game AI

Line Segment Intersection

Line Segment Intersection

- Two notions of intersection
  - Predicate (true/false)
  - Calculation of the actual point of intersection, if it exists
- We will focus on the predicate

Area of a Triangle

- Useful for robust determination of intersection
- From geometry: $A_{Triangle} = \frac{1}{2}bh$
- Vector form: $A_{Triangle} = \frac{1}{2}\|A \times B\|$ (half the magnitude of the Cross Product vector)

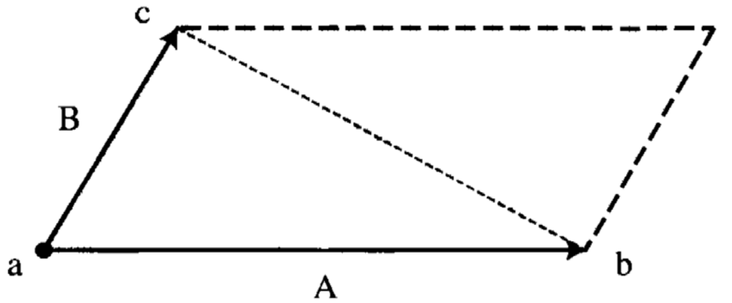
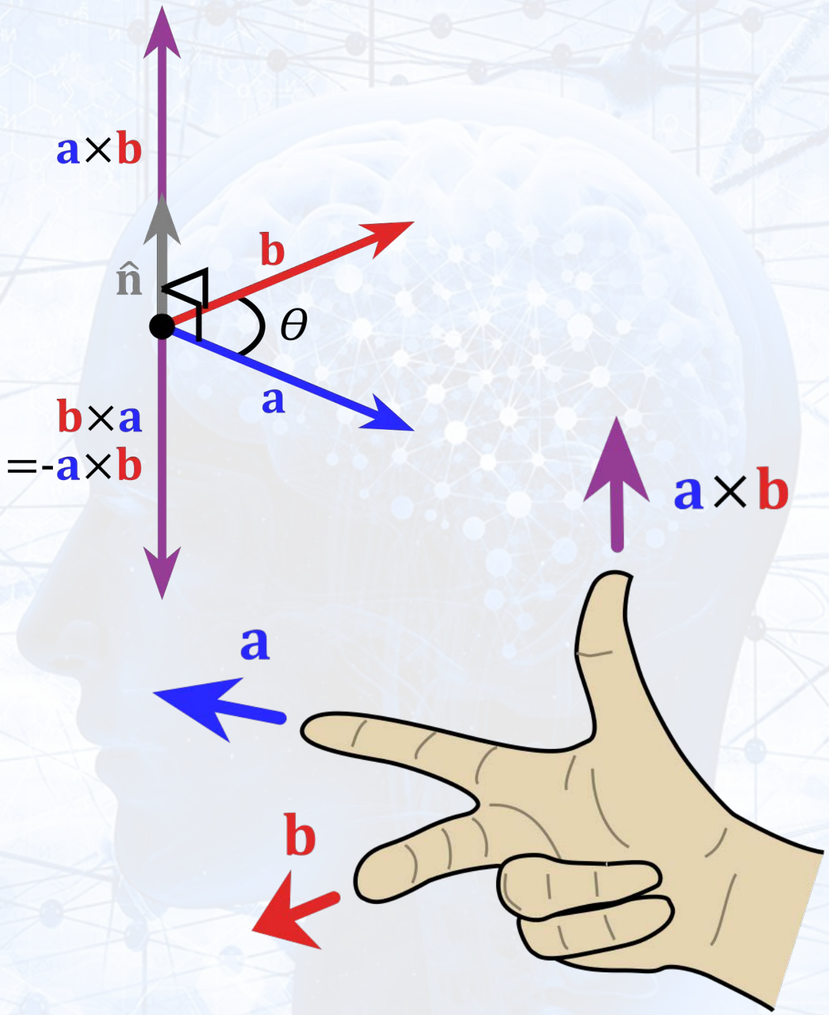


FIGURE 1.15 Cross product parallelogram.

$$\begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ A_0 & A_1 & A_2 \\ B_0 & B_1 & B_2 \end{vmatrix} = (A_1B_2 - A_2B_1)\hat{i} + (A_2B_0 - A_0B_2)\hat{j} + (A_0B_1 - A_1B_0)\hat{k}. \quad (1.1)$$

Cross Product

- Cross product has up or down direction according to righthanded (RHCS) or lefthanded coordinate system (LHCS)
- This direction might be useful…



2D Vector Triangle?

- Cross Product must be 3 dimensions*
- But we can make that work with 2D
- Just plug zero in for *z* value ($\hat{k}$)
- Now cross product result will have *x* and *y* terms zero out, leaving only *z* dimension
- $0\hat{\imath} + 0\hat{\jmath} + (A_0B_1 - A_1B_0)\hat{k}$
- Therefore, the magnitude of the 2D cross product is just the absolute value of the *z* value (scalar of $\hat{k}$)
- But we can leave the sign to indicate the direction

Area of 2D Triangle

- $A_{Triangle} = \frac{1}{2}(A_0B_1 - A_1B_0)$
- Substitute $A = b - a$ and $B = c - a$
- $2A_{Triangle} = a_0b_1 - a_1b_0 + a_1c_0 - a_0c_1 + b_0c_1 - c_0b_1$
  - Move the 2 over to keep result an integer
- Computational Performance improvement
  - Refactor to reduce expensive multiplies (relative to adds)
  - Also, intermediate register values are smaller and less likely to overflow the 32-bit int
- $2A_{Triangle} = (b_0 - a_0)(c_1 - a_1) - (c_0 - a_0)(b_1 - a_1)$

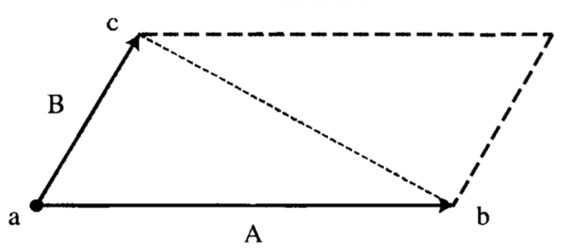


FIGURE 1.15 Cross product parallelogram.

Sign of Area

- If the vertices of the triangle are ordered counterclockwise (CCW) then we can take the area and get a positively signed area with a RHCS
- In this case, *B* is to the Left of *A*
- If *B* is the right of *A*, then the sign of the area is negative
- Furthermore, if the area is 0 then *A* and *B* are collinear (degenerate, flat triangle)

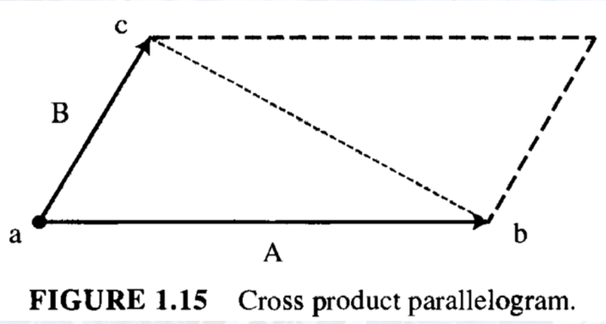


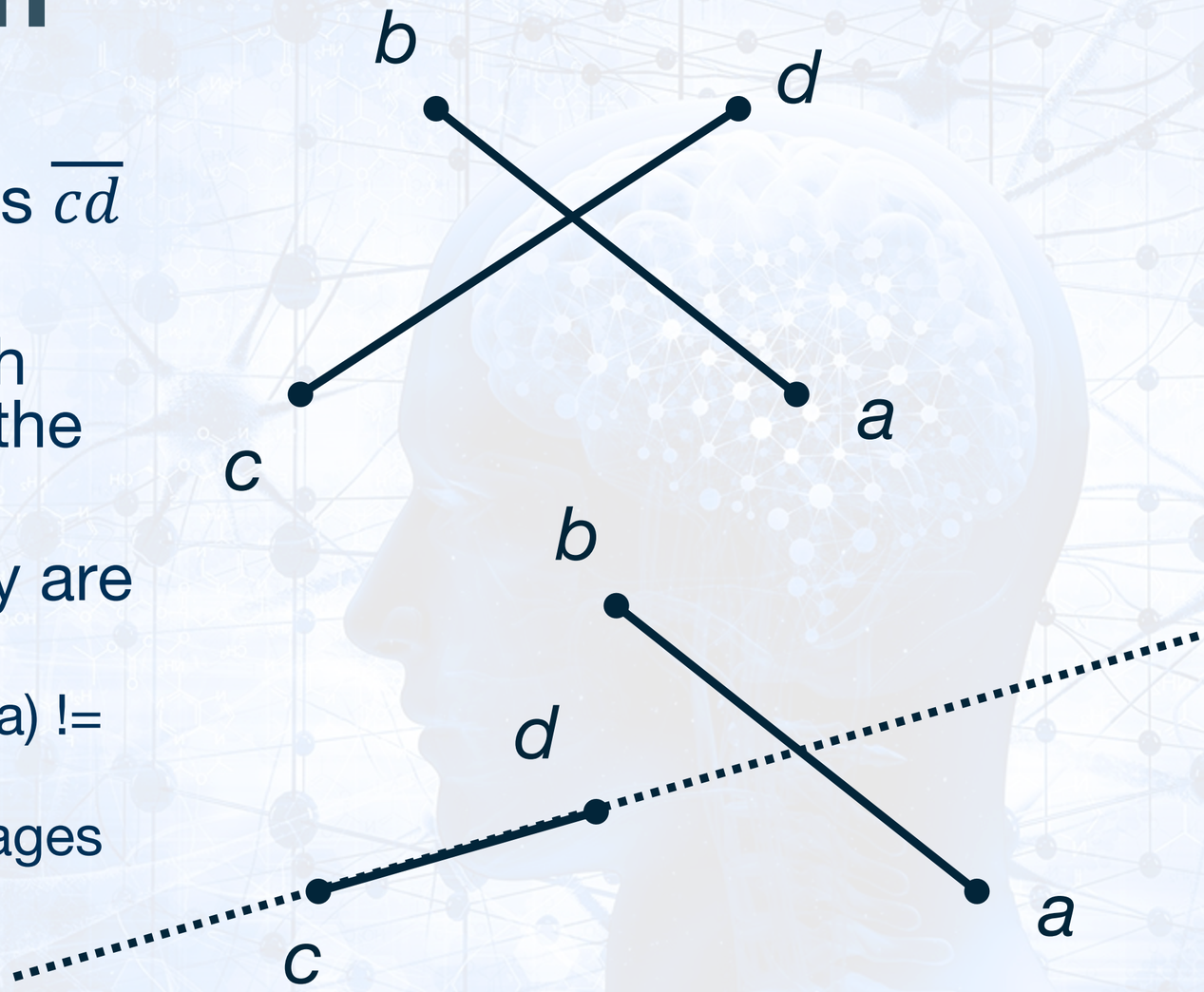
FIGURE 1.15 Cross product parallelogram.

Area-based Predicates

- *Left(a,b,c)* := $2A_{Triangle} > 0$
- *LeftOn(a,b,c)* := $2A_{Triangle} \geq 0$
- *Collinear(a,b,c)* := $2A_{Triangle} == 0$
- These predicates provide a robust way to determine if two line segments intersect!

Boolean Intersection

- An intersection occurs iff $\overline{ab}$ splits $\overline{cd}$ and $\overline{cd}$ splits $\overline{ab}$
- Can use Left() to determine which side each endpoint is relative to the opposing segment
- Four tests total, making sure they are opposites:
  - Left(a,b,c) != Left(a,b,d) && Left(c,d,a) != Left(c,d,b)
  - XOR operation safer in some languages where Boolean is not a first-class primitive



Proper versus Improper Intersection

- We can make the distinction between Proper and Improper intersections with some extensions to the intersection test
- Useful for more complicated tests
- Covered in Computational Geometry in C (2nd Ed.), Joseph O'Rourke (Ch.1)