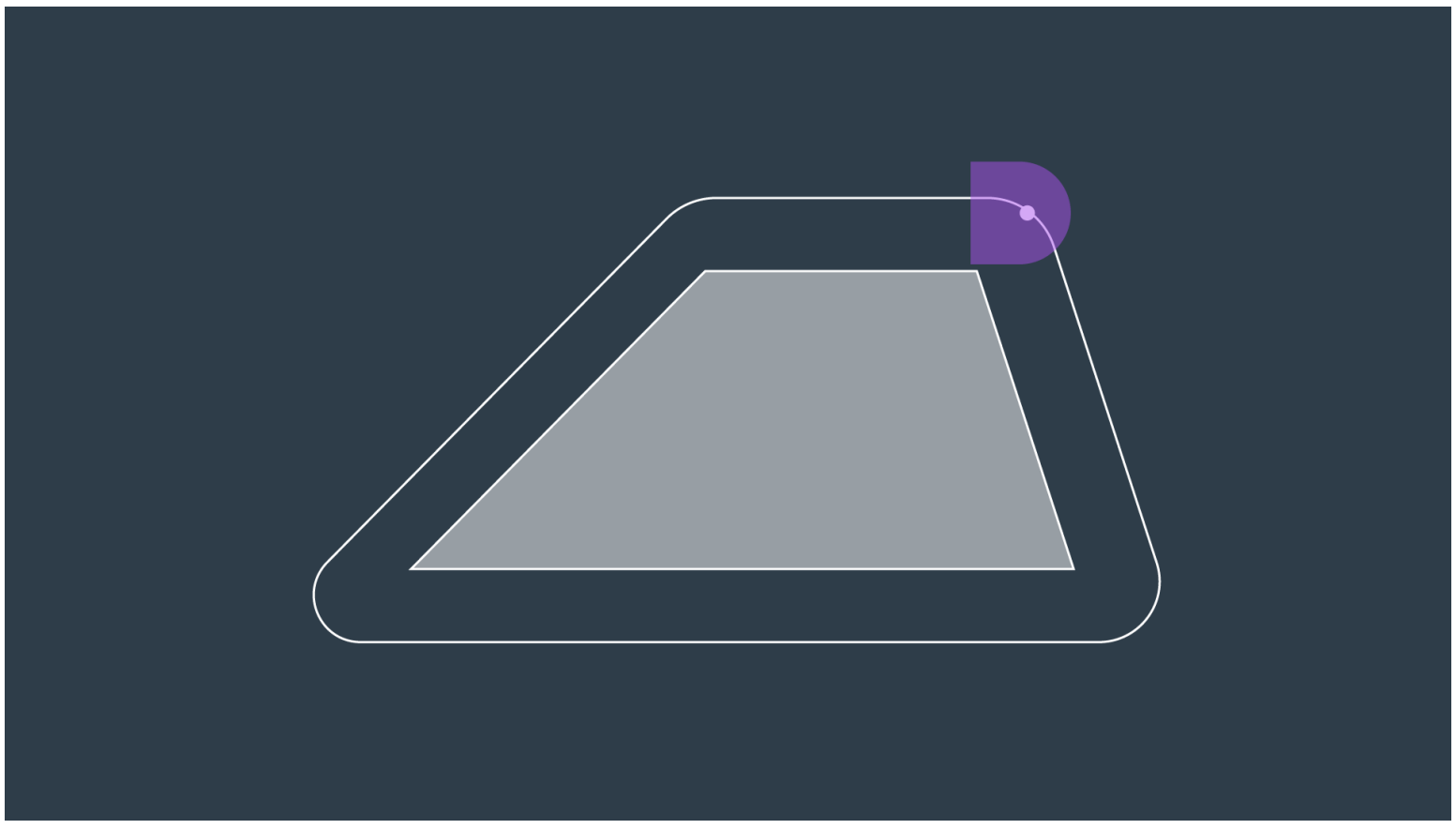
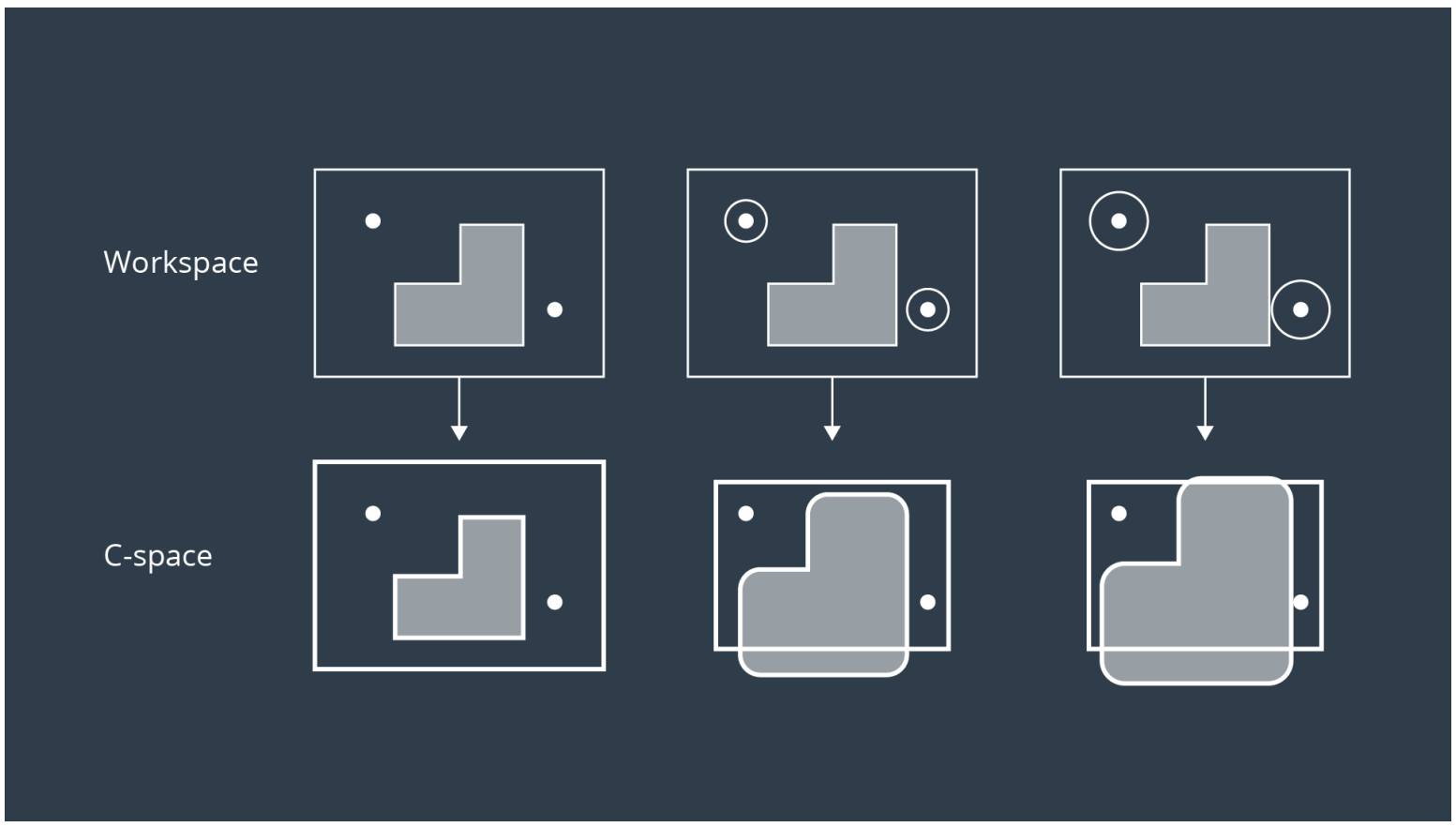
Minkowski Sum

The Minkowski sum is a mathematical property that can be used to compute the configuration space given an obstacle geometry and robot geometry.

The intuition behind how the Minkowski sum is calculated can be understood by imagining to paint the outside of an obstacle using a paintbrush that is shaped like your robot, with the robot's origin as the tip of the paintbrush. The painted area is C_{obs}. The image below shows just this.



To create the configuration space, the Minkowski sum is calculated in such a way for every obstacle in the workspace. The image below shows three configuration spaces created from a single workspace with three different sized robots. As you can see, if the robot is just a dot, then the obstacles in the workspace are only inflated by a small amount to create the C-space. As the size of the robot increases, the obstacles are inflated more and more.



For convex polygons, computing the convolution is trivial and can be done in linear time - however for non-convex polygons (i.e. ones with gaps or holes present), the computation is much more expensive.

Minkowski sums and differences

posted by harrison on September 28, 2012

There are many ways to do collision detection, but a fairly general one is Minkowski differences. The idea is that you do a binary operation on two shapes to get a new shape, and if the origin (the zero vector) is inside that shape, then they are colliding. The minkowski sum lets you define some interesting shapes that are both easy to draw and easy to do collision detection on.

The Minkowski sum

What does it mean to add shapes?

One representation of a shape is a (possibly infinite) set of points. so, a point is just a set with one element, and a circle is the set , or the set of all points within radius of a centre point .

The minkowski sum of and is the set of all points that are the sum of any point in and . The formula is:

Some examples

To instill you with intuition of what a minkowski sum looks like, here are a few examples:

The sum of any shape and a point is that shape translated by that point.

The sum of any shape and two points is two translated (possibly overlapping) copies of that shape.

The sum of two circles is a larger circle (sum the radii) with its centre at the sum of the centres of the smaller circles.

The sum of any shape and a line is that shape swept through that line. Think of placing your shape in sand, and dragging it along the line.

Similarly, the sum of a shape and any curve is what you'd get by sweeping the shape through the curve.

The sum of two parallel lines is a longer line.

For perpendicular lines, you get a square.

Minkowski difference

The minkowski difference is what you get by taking the minkowski sum of a shape and the mirror of another shape. So, your second shape gets flipped about the origin (all of its points are negated).

And if we ask "are these objects colliding?", we're really asking if they have any points in common. If they do have a point in common, then must be in the Minkowski difference. If they do not share a point, then is never , because . So, we have reduced the collision detection problem to a Minkowski sum problem.

Associativity and Commutativity

The Minkowski sum is associative and commutative, so and . This means that we can rearrange sums to make our calculations easier.

So, let's say you have two circles of radius with centre at point and a square of width at centre and at angle , and you want to find . that not so fun, but you can think about it more easily if you rearrange it as , where is a circle at the origin with radius , and is the square centred at the origin. also, you can further decompose into the sum of two line segments, so then you're just sweeping a circle through a line, and then sweeping the resultant capsule through another line, and then you can translate the whole thing.

The Minkowski difference is not commutative, because subtraction is not commutative. It is anticommutative, though, which is just about as good. Any time you flip the order of a difference, you have to negate the result. It's the same as regular subtraction that way.

Convex polygons and lines

This comes up frequently because you will often want to extrude your shapes along their direction of motion, so that if you're checking for collisions at regular intervals, fast-moving objects won't be able to jump over other objects in a single step.

The sum of a polygon and a line is an interesting case, both because it's simple to compute, and because it comes up naturally in collision detection. Suppose you have a polygon represented by line segments, and each line segment has a normal point out. If you take a new line segment , you can split the polygon's edges into three groups: those parallel to , those with their normal pointing the same direction as , or , and those with their normal pointing the opposite direction. If there are no parallel edges, then 2 of the points between edges on the polygon will be between an opposite and a same direction edge. You can insert the new line segment on those points, or extend the existing parallel lines, and then you'll have a new polygon.

This doesn't work quite right with non-convex polygons, because after the operation, the new polygon can intersect itself. You could modify the method to fix the polygon afterward, but it's gross, and not something you want to be doing in your physics calulations.

Higher dimensions

The Minkowski difference method will work in any number of dimensions. There's also a technique to improve on the sweep-test method described above.

Say you're working in 2D, and you have fast-moving objects. Sometimes, they will cross paths in a single time step, but they shouldn't actually collide. if you did a sweep test, they would get marked as having collided, though. One way you can solve this is by going up a dimension to 3D, with time as the third dimension, so instead of extruding in the direction of the velocity, you can extrude in the direction of the velocity with the time component set to 1, instead of 0.

You can cheat a little bit here and bring it back down to a 2D problem by noticing that you only have to worry about the slice of the sum where time = 0. You can rearrange the sum into , and then we can look at , which is a parallelogram, and find the line segment in it where time = 0.

Support functions and line segments

A support function is a function on a set such that, given a direction, you get the point in the set that is furthest in the direction. if there is more than one furthest point, any one of them will do. So, for a circle, you get , and for a rectangle, you get a function that returns one of the corners.

a useful property of the support function is that the support function of a Minkowski sum of shapes is the sum of the support functions of those shapes.

If you have a convex shape with a support function, and you're working in 2D, you can easily find the sum of that shape and a line segment:

find the support points in the directions perpendicular to the line segment (there are two). Then, create a parallelogram from the line segment, and the segment between the two support points. The new shape is the union of the parallelogram, and a copy of the shape at each end of the line segment.

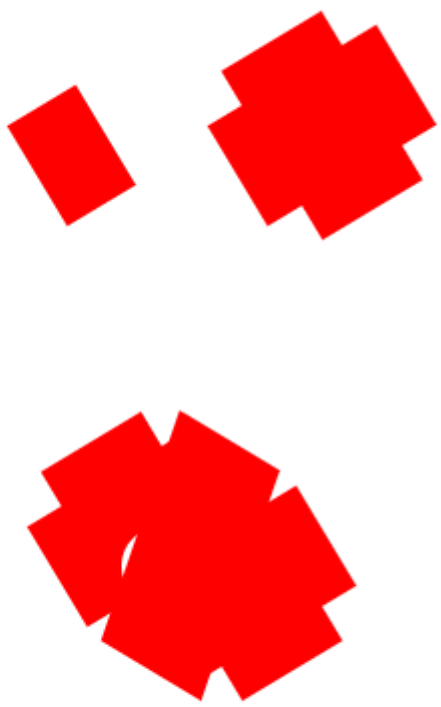
you can also do pretty cool things with Minkowski sums of shapes with support functions and polygons, but i'll leave alone for today.

here's some javascript code as an example of what i mean about line segments: starting with a circle, and adding 1, 2, then 3 lines.

```
var example = document.getElementById('example');
var context = example.getContext('2d');
function draw_parallelogram(l1, l2,x,y) {
    context.beginPath();
    context.moveTo(x+l1.start.x + l2.start.x, y+l1.start.y + l2.start.y);
    context.lineTo(x+l1.start.x + l2.end.x, y+l1.start.y + l2.end.y);
    context.lineTo(x+l1.end.x + l2.end.x, y+l1.end.y + l2.end.y);
    context.lineTo(x+l1.end.x + l2.start.x, y+l1.end.y + l2.start.y);
    context.fill();
}
function draw_circle(c, x, y) {
    context.beginPath();
    context.arc(x+c.x, y + c.y, c.r, 0, 2*3.141592653589793);
    context.fill();
}
function add_line(shape, line) {
    return {
        draw: function(p) {
            shape.draw({x: line.start.x + p.x, y: line.start.y+p.y});
            shape.draw({x: line.end.x + p.x, y: line.end.y + p.y});
            a = shape.support({x:-line.end.y + line.start.y, y:line.end.x - line.start.x});
            b = shape.support({x: line.end.y - line.start.y,y: -line.end.x + line.start.y});
            context.fillStyle = 'red';
            draw_parallelogram({start: a, end: b},line, p.x, p.y);
        },
        support: function(d) {
            s = shape.support(d);
            x = d.x; y = d.y;
            dot = x*(line.end.x-line.start.x) + y*(line.end.y-line.start.y);
```

```
                p = dot > 0 ? line.end : line.start;
                return {x: s.x+p.x, y:s.y+p.y };
            }
        }
    }
    function make_circle(r, x, y) {
        c = {x:x,y:y,r:r};
        return {
            draw: function(p) { context.fillStyle = 'white'; draw_circle(c, p.x, p.y); },
            support: function(d) {
                l = Math.sqrt(d.x*d.x+d.y*d.y);
                return {x:d.x/l * r+x, y: d.y/l * r + y};
            }
        }
    }
    circ = make_circle(20, 0,0);
    capsule = add_line(circ, {start: {x:0,y:0}, end:{x:30,y:50}});
    round_square = add_line(capsule, {start: {x:0,y:0}, end:{x:50,y:-30}});
    hex = add_line(round_square, {start:{x:10,y:0}, end:{x:60,y:30}});
    circ.draw({x:300,y:300});
    capsule.draw({x:100,y:100});
    round_square.draw({x:200,y:100});
    hex.draw({x:100,y:300});
```

and the output:



Comments are closed.

Twisted Oak Studios offers consulting and development on high-tech interactive projects. Check out our portfolio, or Give us a shout if you have anything you think some really rad engineers should help you with.

Archive

- Implementing Quantum Pseudo-Telepathy
- Turn On Your Damn Warnings
- Big-O Made Trivial
- Unfathomable Bugs #7: The Broken Oven
- Solomonoff's Mad Scientist
- Yearly Blogging Roundup #1
- What isn't a Monad
- Searching a Sorted Matrix Faster
- How to Read Nested Ternary Operators
- Making Sublime Text 2 Jump to the Correct Line with Unity on OS X
- My Bug, My Bad #4: Reading Concurrently
- Whole API Testing with Reflection
- Optimizing a Parser Combinator into a memcpy
- Don't Treat Paths Like Strings
- Breaking a Toy Hash Function
- Counting Iterators Lazily
- Unfathomable Bugs #6: Pretend Precision
- My Bug, My Bad #3: Accidentally Attacking WarCraft 3
- Collapsing Types vs Monads (followup)
- Collapsing Futures: Easy to Use, Hard to Represent
- Eventual Exceptions vs Programming in a Minimal Functional Style
- The Mystery of Flunf
- Explain it like I'm Five: The Socialist Millionaire Problem and Secure Multi-Party Computation
- Computer Science Blows My Mind
- A visit to Execution Labs in Montréal
- Transmuting Dice, Conserving Entropy
- Rule of Thumb: Ask for the Clock
- Rule of Thumb: Use Purposefully Weakened Methods
- Rule of thumb: Preconditions Should be Checked Explicitly
- Intersecting Linked Lists Faster
- Mouse Path Smoothing for Jack Lumber
- My Bug, My Bad #2: Sunk by Float
- Repeat Yourself Differently
- Grover's Quantum Search Algorithm
- Followup to Non-Nullable Types vs C#
- Optimizing Just in Time with Expression Trees
- When One-Way Latency Doesn't Matter
- Determining exactly if/when/where a moving line intersected a moving point
- Emulating Actors in C# with Async/Await
- Making an immutable queue with guaranteed constant time operations
- Improving Checked Exceptions
- Perishable Collections: The Benefits of Removal-by-Lifetime
- Decoupling shared control
- Decoupling inlined UI code
- Linq to Collections: Beyond IEnumerable<T>
- Publish your .Net library as a NuGet package
- When null is not enough: an option type for C#
- Unfathomable Bugs #5: Readonly or not
- Minkowski sums: examples
- My Bug, My Bad #1: Fractal Spheres
- Working around the brittle UI Virtualization in Windows 8
- Encapsulating Angles
- Unfathomable Bugs #4: Keys that aren't
- How would I even use a monad (in C#)?
- Useful/Interesting Methods #1: Observable.WhenEach
- Unfathomable Bugs #3: Stringing you along
- Anonymous Implementation Classes – A Design Pattern for C#
- Tasks for ActionScript 3 – Improving on Event-Driven Programming
- Minkowski sums and differences
- Non-Nullable Types vs C#: Fixing the Billion Dollar Mistake
- Unfathomable Bugs #2: Slashing Out
- Script templates and base classes
- Unity font extraction
- Abusing "Phantom Types" to Encode List Lengths Into Their Type
- Constructive Criticism of the Reactive Extensions API
- Quaternions part 3
- Quaternions part 2
- Quaternions part 1
- Unfathomable Bugs #1: You can have things! You can have things IN things! You can have …
- Coroutines – More than you want to know
- Asset Bundle Helper
- The Visual Studio goes away
- .Net's time traveling StopWatch
- Introducing Catalyst

twisted oak

contact@twistedoak.co

WIKIPEDIA

Minkowski addition

In geometry, the **Minkowski sum** (also known as dilation) of two sets of position vectors A and B in Euclidean space is formed by adding each vector in A to each vector in B, i.e., the set

$$A + B = \{\mathbf{a} + \mathbf{b} \mid \mathbf{a} \in A,\ \mathbf{b} \in B\}.$$

Analogously, the **Minkowski difference** (or geometric difference)[1] is defined as

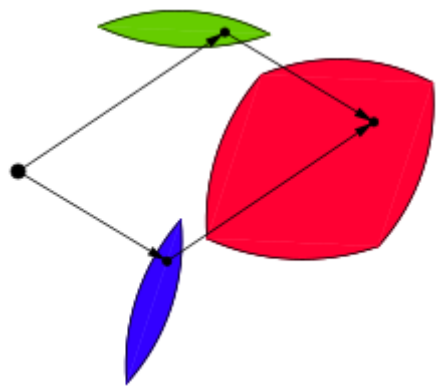
$$A - B = \{\mathbf{c} \mid \mathbf{c} + B \subseteq A\}.$$

It is important to note that in general $A - B \neq A + (-B)$. For instance, in a one-dimensional case $A = [-2, 2]$ and $B = [-1, 1]$ the Minkowski difference $A - B = [-1, 1]$, whereas $A + (-B) = A + B = [-3, 3]$. The correct formula connecting Minkowski sum and difference is as follows (here X^c stands for the complement of X):

$$A - B = (A^c + (-B))^c.$$

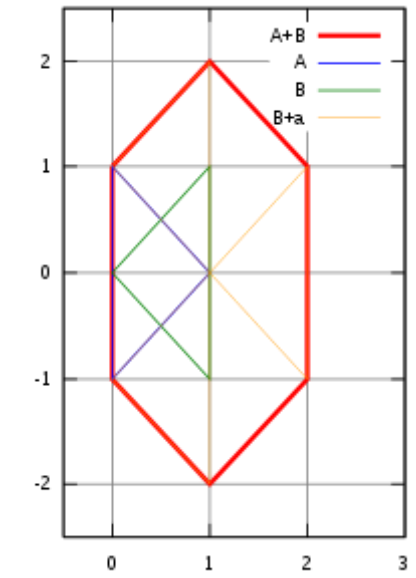
In a two-dimensional case, Minkowski difference is closely related to erosion (morphology) in image processing.

The concept is named for Hermann Minkowski.



The red figure is the Minkowski sum of blue and green figures.

Contents



Minkowski sum $A + B$

Example

For example, if we have two sets A and B, each consisting of three position vectors (informally, three points), representing the vertices of two triangles in $\mathbb{R}^2$, with coordinates

$$A = \{(1, 0), (0, 1), (0, -1)\}$$

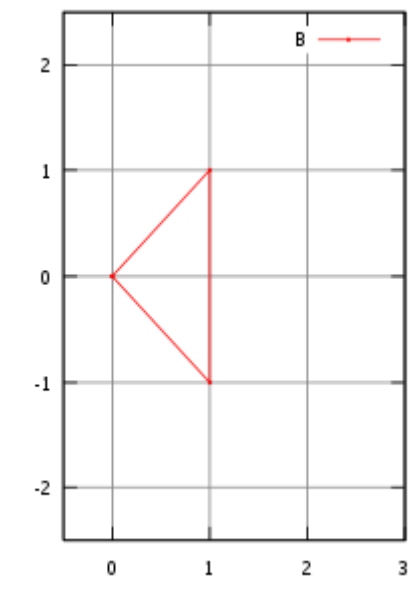
and

$$B = \{(0, 0), (1, 1), (1, -1)\},$$

then their Minkowski sum is

$$A + B = \{(1, 0), (2, 1), (2, -1), (0, 1), (1, 2), (0, -1), (1, -2)\},$$

which comprises the vertices of a hexagon.



B

For Minkowski addition, the *zero set*, $\{0\}$, containing only the zero vector, 0, is an identity element: for every subset S of a vector space,

$$S + \{0\} = S.$$

The empty set is important in Minkowski addition, because the empty set annihilates every other subset: for every subset S of a vector space, its sum with the empty set is empty:

$$S + \emptyset = \emptyset.$$

Convex hulls of Minkowski sums

Minkowski addition behaves well with respect to the operation of taking convex hulls, as shown by the following proposition:

- For all non-empty subsets S_1 and S_2 of a real vector space, the convex hull of their Minkowski sum is the Minkowski sum of their convex hulls:

$$\mathbf{Conv}(S_1 + S_2) = \mathbf{Conv}(S_1) + \mathbf{Conv}(S_2).$$

This result holds more generally for any finite collection of non-empty sets:

$$\mathbf{Conv}\left(\sum S_n\right) = \sum \mathbf{Conv}(S_n).$$

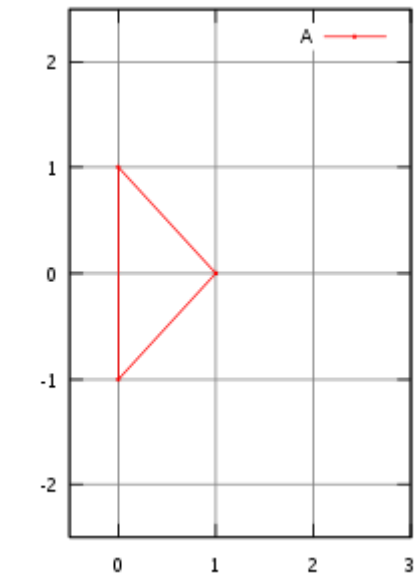
In mathematical terminology, the operations of Minkowski summation and of forming convex hulls are commuting operations.[2][3]

If S is a convex set then $\mu S + \lambda S$ is also a convex set; furthermore

$$\mu S + \lambda S = (\mu + \lambda) S$$

for every $\mu, \lambda \geq 0$. Conversely, if this "distributive property" holds for all non-negative real numbers, μ, λ, then the set is convex.[4]

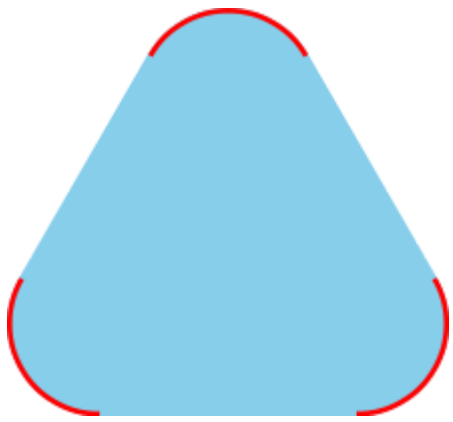
The figure shows an example of a non-convex set for which $A + A \supsetneq 2A$.



A

An example in 1 dimension is: B=[1,2]∪[4,5]. It can be easily calculated that $2B$=[2,4]∪[8,10] but B+B=[2,4]∪[5,7]∪[8,10], hence again B+$B \supsetneq 2B$.

Minkowski sums act linearly on the perimeter of two-dimensional convex bodies: the perimeter of the sum equals the sum of perimeters. Additionally, if K is (the interior of) a curve of constant width, then the Minkowski sum of K and of its 180° rotation is a disk. These two facts can be combined to give a short proof of Barbier's theorem on the perimeter of curves of constant width.[5]



In the convex hull of the red set, each blue point is a convex combination of some red points.

Applications

Minkowski addition plays a central role in mathematical morphology. It arises in the brush-and-stroke paradigm of 2D computer graphics (with various uses, notably by Donald E. Knuth in Metafont), and as the solid sweep operation of 3D computer graphics.

Motion planning

Minkowski sums are used in motion planning of an object among obstacles. They are used for the computation of the configuration space, which is the set of all admissible positions of the object. In the simple model of translational motion of an object in the plane, where the position of an object may be uniquely specified by the position of a fixed point of this object, the configuration space are the Minkowski sum of the set of obstacles and the movable object placed at the origin and rotated 180 degrees.

Numerical control (NC) machining

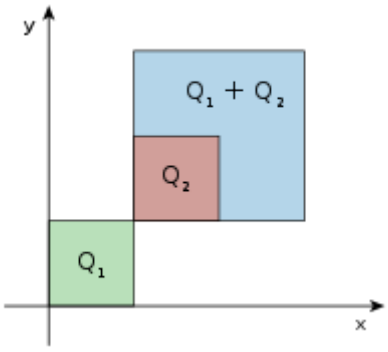
In numerical control machining, the programming of the NC tool exploits the fact that the Minkowski sum of the cutting piece with its trajectory gives the shape of the cut in the material.

3d Solid Modeling

In OpenSCAD Minkowski sums are used to outline a shape with another shape creating a composite of both shapes.

Aggregation Theory

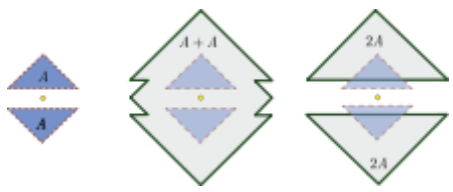
Minkowski sums are also frequently used in aggregation theory when individual objects to be aggregated are characterized via sets.[6][7]



Minkowski addition of sets. The sum of the squares Q_1=$[0,1]^2$ and Q_2=$[1,2]^2$ is the square Q_1+Q_2=$[1,3]^2$.

Algorithms for computing Minkowski sums

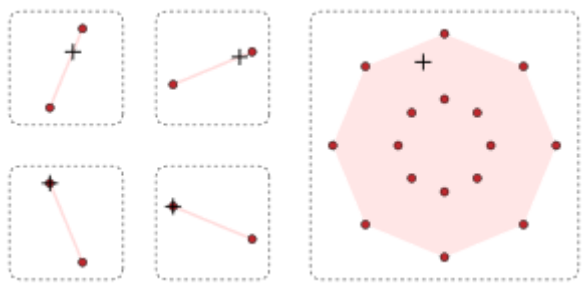
Planar case



An example of a non-convex set such that $A + A \neq 2A$

Two convex polygons in the plane

For two convex polygons P and Q in the plane with m and n vertices, their Minkowski sum is a convex polygon with at most m + n vertices and may be computed in time O (m + n) by a very simple procedure, which may be informally described as follows. Assume that the edges of a polygon are given and the direction, say, counterclockwise, along the polygon boundary. Then it is easily seen that these edges of the convex polygon are ordered by polar angle. Let us merge the ordered sequences of the directed edges from P and Q into a single ordered sequence S. Imagine that these edges are solid arrows which can be moved freely while keeping them parallel to their original direction. Assemble these arrows in the order of the sequence S by attaching the tail of the next arrow to the head of the previous arrow. It turns out that the resulting polygonal chain will in fact be a convex polygon which is the Minkowski sum of P and Q.



Minkowski addition and convex hulls. The sixteen dark-red points (on the right) form the Minkowski sum of the four non-convex sets (on the left), each of which consists of a pair of red points. Their convex hulls (shaded pink) contain plus-signs (+): The right plus-sign is the sum of the left plus-signs.

Other

If one polygon is convex and another one is not, the complexity of their Minkowski sum is O(nm). If both of them are nonconvex, their Minkowski sum complexity is O((mn)^2).

Essential Minkowski sum

There is also a notion of the **essential Minkowski sum** $+_e$ of two subsets of Euclidean space. Note that the usual Minkowski sum can be written as

$$A + B = \{z \in \mathbb{R}^n \mid A \cap (z - B) \neq \emptyset\}.$$

Thus, the **essential Minkowski sum** is defined by

$$A +_{\mathrm{e}} B = \{z \in \mathbb{R}^n \,|\, \mu\,[A \cap (z - B)] > 0\},$$

where μ denotes the n-dimensional Lebesgue measure. The reason for the term "essential" is the following property of indicator functions: while

$$1_{A+B}(z) = \sup_{x \in \mathbb{R}^n} 1_A(x) 1_B(z - x),$$

it can be seen that

$$1_{A +_{\mathrm{e}} B}(z) = \operatorname*{ess\,sup}_{x \in \mathbb{R}^n} 1_A(x) 1_B(z - x),$$

where "ess sup" denotes the essential supremum.

L^p Minkowski sum

For K and L compact convex subsets in $\mathbb{R}^n$, the Minkowski sum can be described by the support function of the convex sets:

$$h_{K+L} = h_K + h_L.$$

For $p \geq 1$, Firey[8] defined the **$\mathbf{L^p}$ Minkowski sum** $K +_p L$ of compact convex sets K and L in $\mathbb{R}^n$ containing the origin as

$$h^p_{K +_p L} = h^p_K + h^p_L.$$

By the Minkowski inequality, the function $h_{K +_p L}$ is again positive homogeneous and convex and hence the support function of a compact convex set. This definition is fundamental in the L^p Brunn-Minkowski theory.

See also

- Dilation
- Erosion
- Interval arithmetic
- Mixed volume (a.k.a. Quermassintegral or intrinsic volume)
- Parallel curve
- Shapley–Folkman lemma
- Zonotope
- Convolution

Notes

1. Hadwiger, Hugo (1950), "Minkowskische Addition und Subtraktion beliebiger Punktmengen und die Theoreme von Erhard Schmidt", *Math. Z.*, **53** (3): 210–218, doi:10.1007/BF01175656 (https://doi.org/10.1007%2FBF01175656)
2. Theorem 3 (pages 562–563): Krein, M.; Šmulian, V. (1940). "On regularly convex sets in the space conjugate to a Banach space". *Annals of Mathematics*. Second Series. **41**. pp. 556–583. doi:10.2307/1968735 (https://doi.org/10.2307%2F1968735). JSTOR 1968735 (https://www.jstor.org/stable/1968735). MR 0002009 (https://www.ams.org/mathscinet-getitem?mr=0002009).
3. For the commutativity of Minkowski addition and convexification, see Theorem 1.1.2 (pages 2–3) in Schneider; this reference discusses much of the literature on the convex hulls of Minkowski sumsets in its "Chapter 3 Minkowski addition" (pages 126–196): Schneider, Rolf (1993). *Convex bodies: The Brunn–Minkowski theory*. Encyclopedia of mathematics and its applications. **44**. Cambridge: Cambridge University Press. pp. xiv+490. ISBN 978-0-521-35220-8. MR 1216521 (https://www.ams.org/mathscinet-getitem?mr=1216521).
4. Chapter 1: Schneider, Rolf (1993). *Convex bodies: The Brunn–Minkowski theory*. Encyclopedia of mathematics and its applications. **44**. Cambridge: Cambridge University Press. pp. xiv+490. ISBN 978-0-521-35220-8. MR 1216521 (https://www.ams.org/mathscinet-getitem?mr=1216521).
5. The Theorem of Barbier (Java) (http://www.cut-the-knot.org/ctk/Barbier.shtml) at cut-the-knot.
6. Zelenyuk, V (2015). "Aggregation of scale efficiency" (https://ideas.repec.org/a/eee/ejores/v240y2015i1p269-277.html). *European Journal of Operational Research*. **240** (1): 269–277. doi:10.1016/j.ejor.2014.06.038 (https://doi.org/10.1016%2Fj.ejor.2014.06.038).
7. Mayer, A.; Zelenyuk, V. (2014). "Aggregation of Malmquist productivity indexes allowing for reallocation of resources" (https://ideas.repec.org/a/eee/ejores/v238y2014i3p774-785.html). *European Journal of Operational Research*. **238** (3): 774–785. doi:10.1016/j.ejor.2014.04.003 (https://doi.org/10.1016%2Fj.ejor.2014.04.003).
8. Firey, William J. (1962), "*p*-means of convex bodies", *Math. Scand.*, **10**: 17–24, doi:10.7146/math.scand.a-10510 (https://doi.org/10.7146%2Fmath.scand.a-10510)

References

- Arrow, Kenneth J.; Hahn, Frank H. (1980). *General competitive analysis*. Advanced textbooks in economics. **12** (reprint of (1971) San Francisco, CA: Holden-Day, Inc. Mathematical economics texts. **6** ed.). Amsterdam: North-Holland. ISBN 978-0-444-85497-1. MR 0439057 (https://www.ams.org/mathscinet-getitem?mr=0439057).
- Gardner, Richard J. (2002), "The Brunn-Minkowski inequality", *Bull. Amer. Math. Soc. (N.S.)*, **39** (3): 355–405 (electronic), doi:10.1090/S0273-0979-02-00941-2 (https://doi.org/10.1090%2FS0273-0979-02-00941-2)
- Green, Jerry; Heller, Walter P. (1981). "1 Mathematical analysis and convexity with applications to economics" (http://www.sciencedirect.com/science/article/B7P5Y-4FDF0FN-5/2/613440787037f7f62d65a05172503737). In Arrow, Kenneth Joseph; Intriligator, Michael D (eds.). *Handbook of mathematical economics, Volume **I***. Handbooks in economics. **1**. Amsterdam: North-Holland Publishing Co. pp. 15–52. doi:10.1016/S1573-4382(81)01005-9 (https://doi.org/10.1016%2FS1573-4382%2881%2901005-9). ISBN 978-0-444-86126-9. MR 0634800 (https://www.ams.org/mathscinet-getitem?mr=0634800).
- Henry Mann (1976), *Addition Theorems: The Addition Theorems of Group Theory and Number Theory* (Corrected reprint of 1965 Wiley ed.), Huntington, New York: Robert E. Krieger Publishing Company, ISBN 978-0-88275-418-5 – via http://www.krieger-publishing.com/subcats/MathematicsandStatistics/mathematicsandstatistics.html
- Rockafellar, R. Tyrrell (1997). *Convex analysis*. Princeton landmarks in mathematics (Reprint of the 1979 Princeton mathematical series **28** ed.). Princeton, NJ: Princeton University Press. pp. xviii+451. ISBN 978-0-691-01586-6. MR 1451876 (https://www.ams.org/mathscinet-getitem?mr=1451876).
- Nathanson, Melvyn B. (1996), *Additive Number Theory: Inverse Problems and Geometry of Sumsets*, GTM, **165**, Springer, Zbl 0859.11003 (https://zbmath.org/?format=complete&q=an:0859.11003).
- Oks, Eduard; Sharir, Micha (2006), "Minkowski Sums of Monotone and General Simple Polygons", *Discrete and Computational Geometry*, **35** (2): 223–240, doi:10.1007/s00454-005-1206-y (https://doi.org/10.1007%2Fs00454-005-1206-y).
- Schneider, Rolf (1993), *Convex bodies: the Brunn-Minkowski theory*, Cambridge: Cambridge University Press.

- Tao, Terence & Vu, Van (2006), *Additive Combinatorics*, Cambridge University Press.
- Mayer, A.; Zelenyuk, V. (2014). "Aggregation of Malmquist productivity indexes allowing for reallocation of resources" (https://ideas.repec.org/a/eee/ejores/v238y2014i3p774-785.html). *European Journal of Operational Research*. **238** (3): 774–785. doi:10.1016/j.ejor.2014.04.003 (https://doi.org/10.1016%2Fj.ejor.2014.04.003).
- Zelenyuk, V (2015). "Aggregation of scale efficiency" (https://ideas.repec.org/a/eee/ejores/v240y2015i1p269-277.html). *European Journal of Operational Research*. **240** (1): 269–277. doi:10.1016/j.ejor.2014.06.038 (https://doi.org/10.1016%2Fj.ejor.2014.06.038).

External links

- Hazewinkel, Michiel, ed. (2001) [1994], "Minkowski addition" (https://www.encyclopediaofmath.org/index.php?title=p/m120210), *Encyclopedia of Mathematics*, Springer Science+Business Media B.V. / Kluwer Academic Publishers, ISBN 978-1-55608-010-4
- Howe, Roger (1979), *On the tendency toward convexity of the vector sum of sets* (http://econpapers.repec.org/RePEc:cwl:cwldpp:538), Cowles Foundation discussion papers, **538**, Cowles Foundation for Research in Economics, Yale University
- Minkowski Sums (http://www.cgal.org/Pkg/MinkowskiSum2), in Computational Geometry Algorithms Library
- The Minkowski Sum of Two Triangles (http://demonstrations.wolfram.com/TheMinkowskiSumOfTwoTriangles/) and The Minkowski Sum of a Disk and a Polygon (http://demonstrations.wolfram.com/TheMinkowskiSumOfADiskAndAPolygon/) by George Beck, The Wolfram Demonstrations Project.
- Minkowski's addition of convex shapes (http://www.cut-the-knot.org/Curriculum/Geometry/PolyAddition.shtml) by Alexander Bogomolny: an applet
- Wikibooks:OpenSCAD User Manual/Transformations#minkowski by Marius Kintel: Application