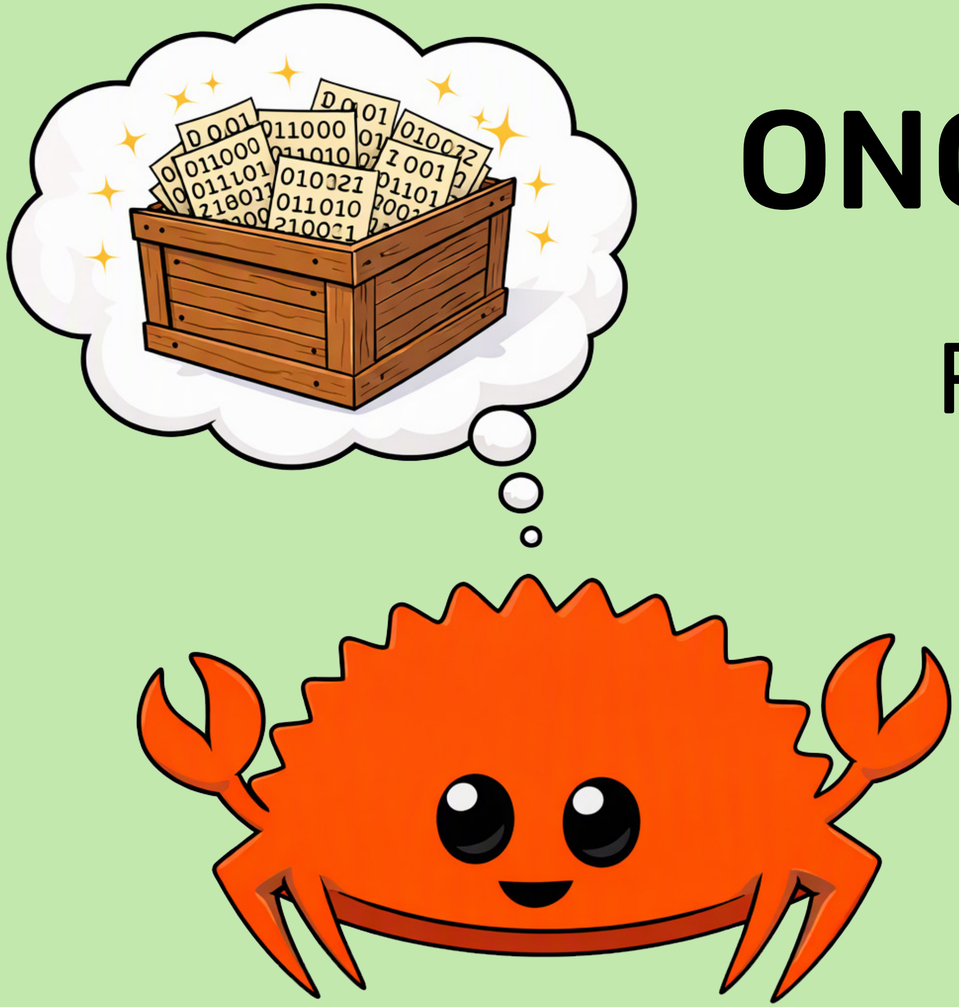
MICHEL KRÄMER

Researcher @ Fraunhofer

CAN YOU FIND RUST ON A MAP?

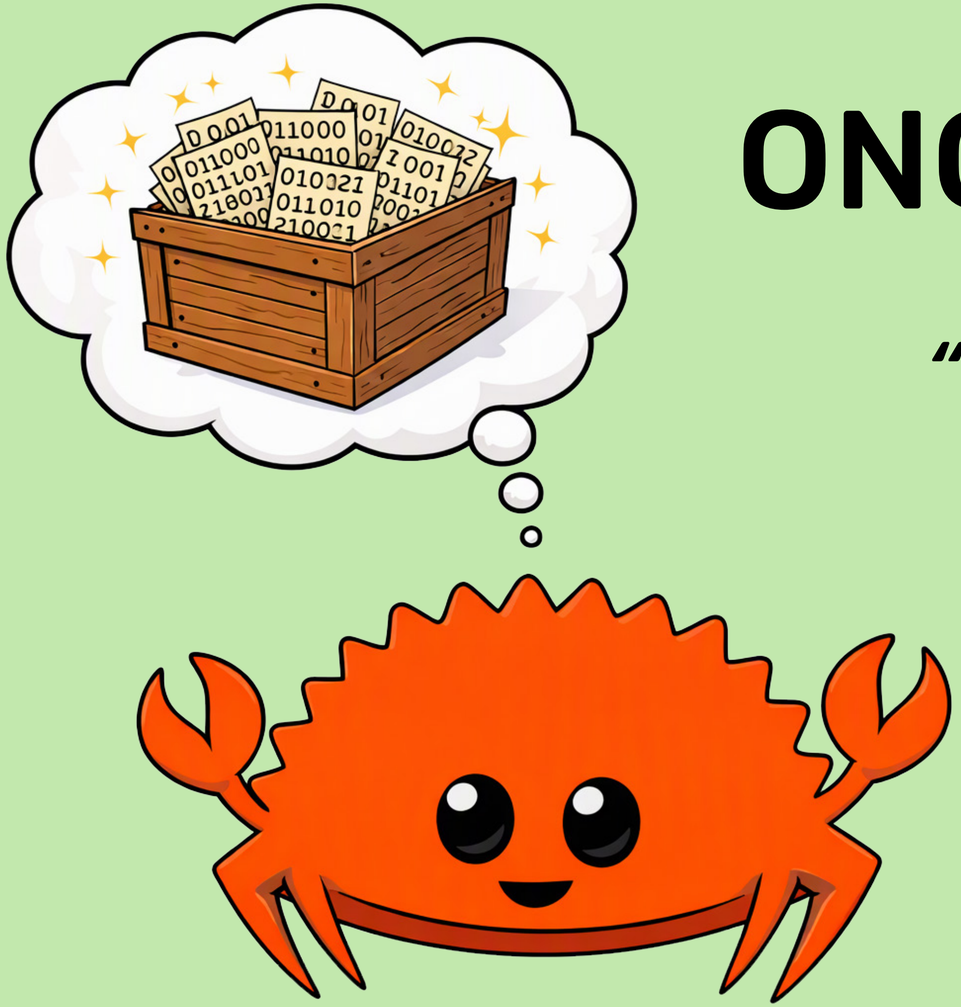


RUSTLAB 2026



ONCE UPON A TIME ...

Ferris was looking for
a crate full of
hand-written code.



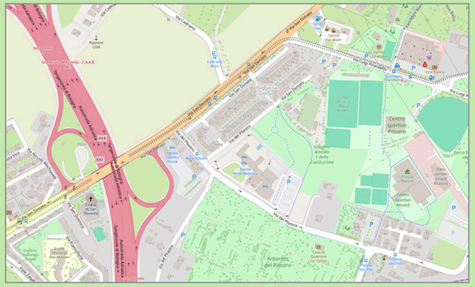
ONCE UPON A TIME ...

“Something like this is hard to come by these days,” he said.

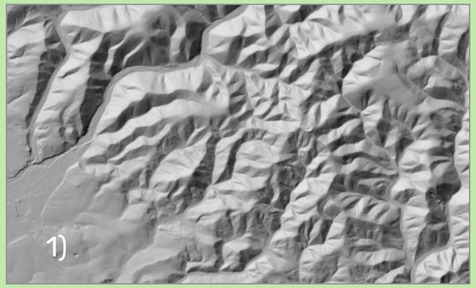
BUT WHERE SHOULD HE LOOK?



HE WAS AN EXPERT IN **GEOSPATIAL DATA**



Vector data



Raster data



Imagery



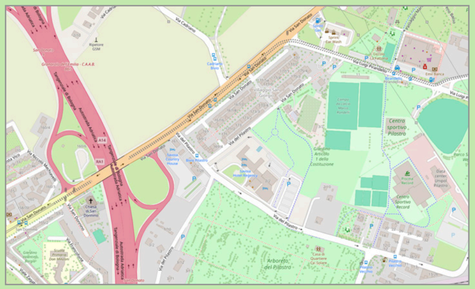
Point clouds



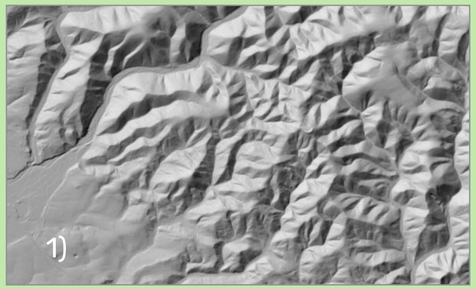
More: metadata, time series, ...

1) Geobasisdaten: © GeoBasis-DE / BKG (2026)
2) © Bundesamt für Kartographie und Geodäsie, Hexagon (2021)

HE WAS AN EXPERT IN **GEOSPATIAL DATA**



Vector data



Raster data



Imagery



Point clouds



More: metadata, time series, ...

1) Geobasisdaten: © GeoBasis-DE / BKG (2026)
2) © Bundesamt für Kartographie und Geodäsie, Hexagon (2021)

HE WAS AN EXPERT IN **GEOSPATIAL OPERATIONS**

- Spatial queries
- Coordinate transformations
- Boolean and buffer operations
- AI-based image analysis



And many more ...

HE WAS AN EXPERT IN **GEOSPATIAL OPERATIONS**

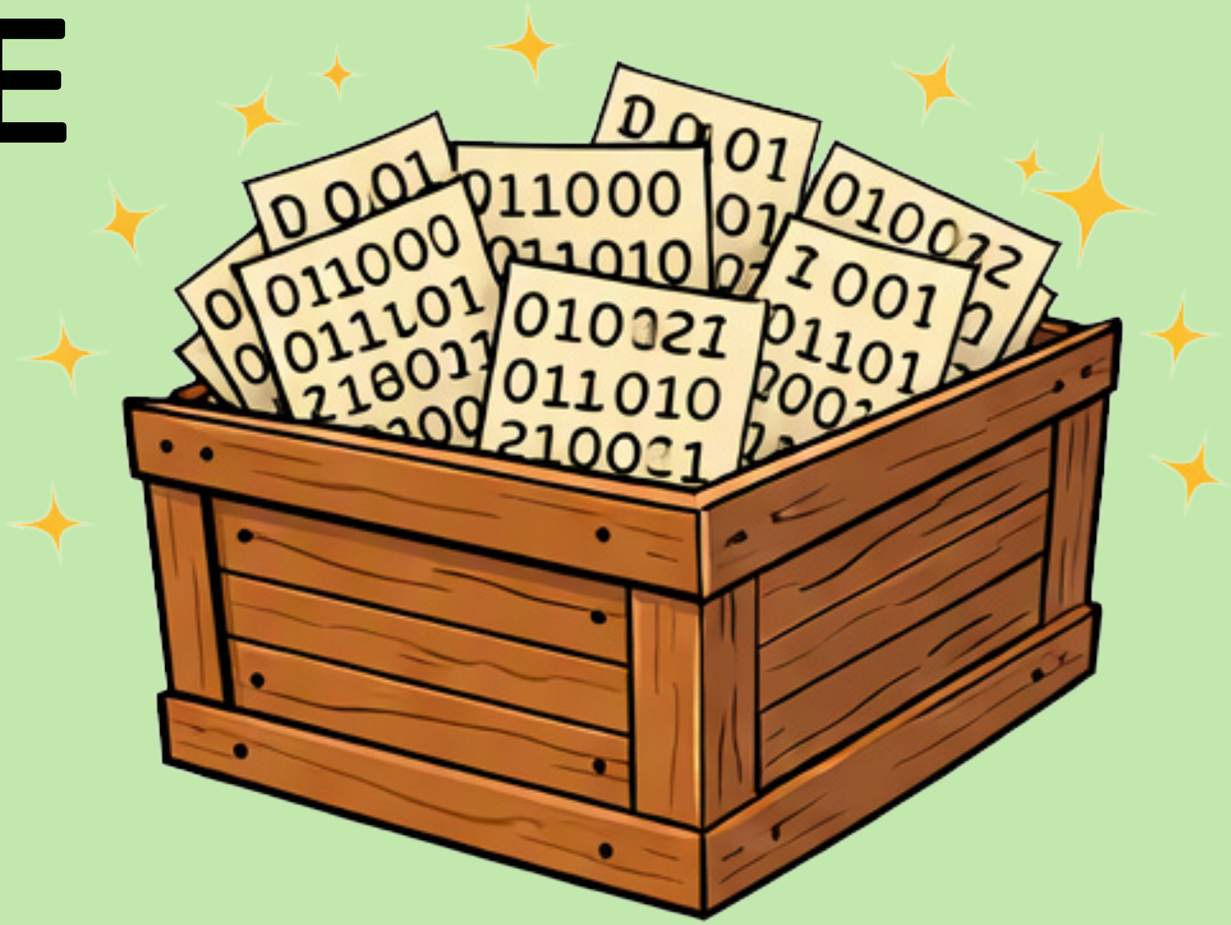
- Spatial queries
- Coordinate transformations
- Boolean and buffer operations
- AI-based image analysis

And many more ...



Chapter I

THE CRATE FULL OF CODE



FACTS HE KNEW ABOUT THE CRATE

FACTS HE KNEW ABOUT THE CRATE

1. Somewhere in Europe

FACTS HE KNEW ABOUT THE CRATE

1. Somewhere in Europe

2. Guarded by pirates

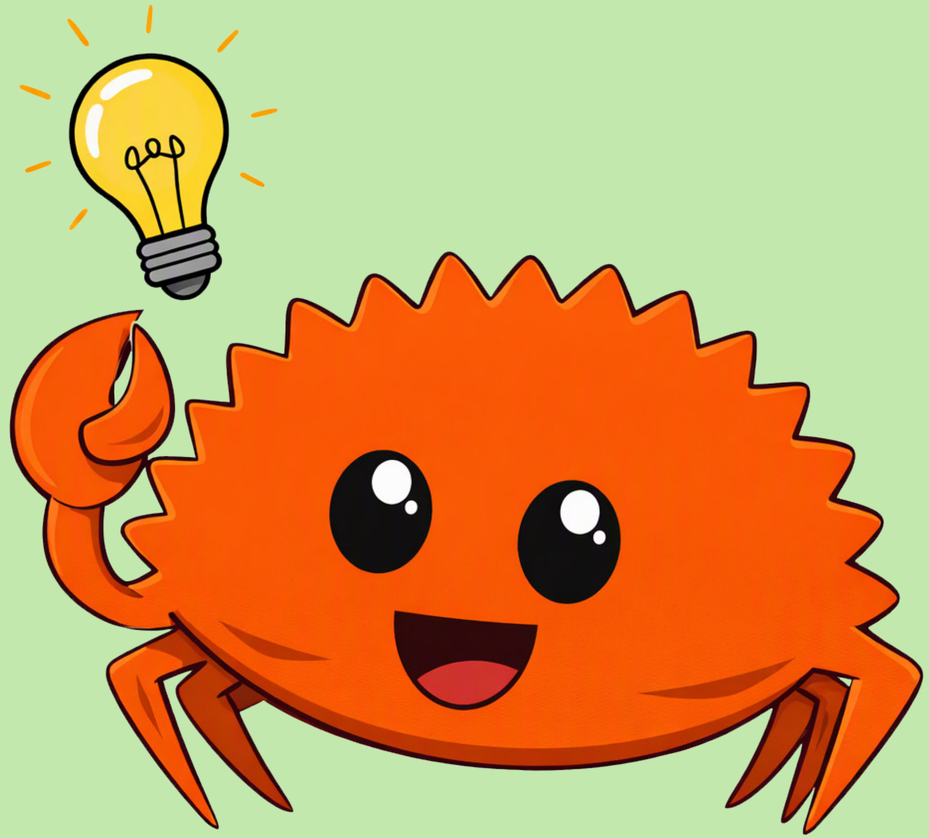
FACTS HE KNEW ABOUT THE CRATE

1. Somewhere in Europe
2. Guarded by pirates
3. Pirates love ice cream



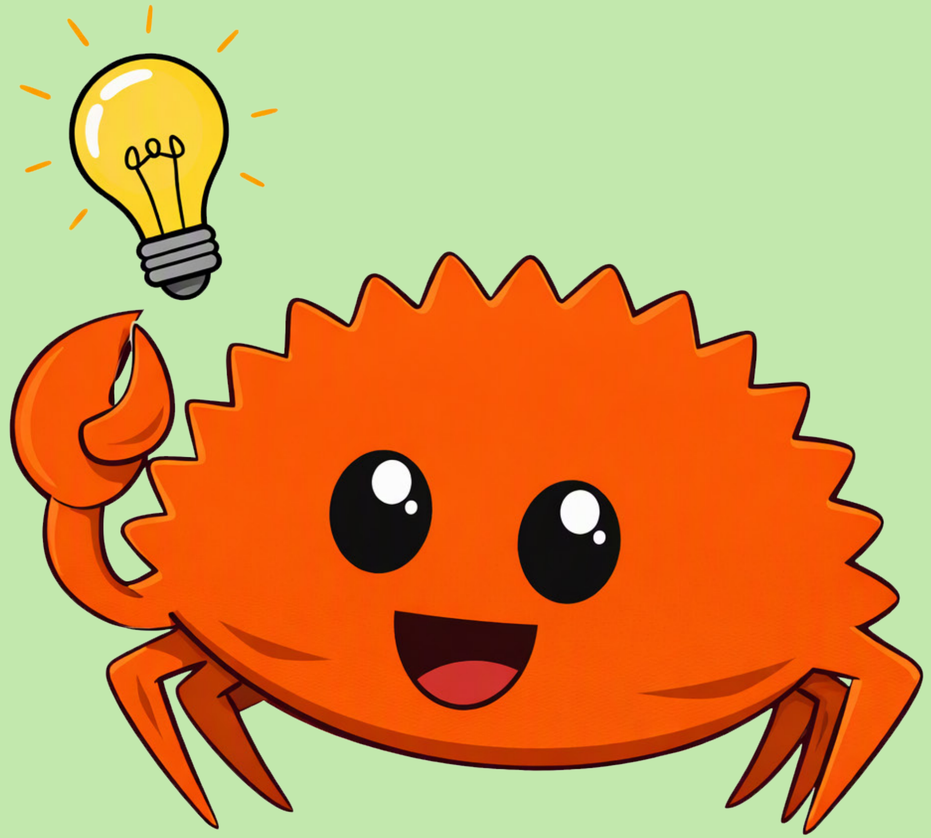
THE **QUESTION** HE ASKED

1. “Find a **place** in **Europe**
2. with a **pirate den** that is
3. very close to an **ice cream parlor**”



THE **DATA** HE NEEDED

1. Places (names)
2. Possible pirate dens
3. Ice cream parlors



WHERE TO GET DATA

- OpenStreetMap
  https://www.openstreetmap.org

- https://download.geofabrik.de
  File format: .osm.pbf
  World: 80 GB, Europe: 33 GB

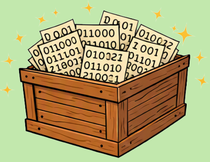


READING .OSM.PBF IN RUST

See also: https://wiki.openstreetmap.org/wiki/Rust



osmpbf



osmpbfreader

My experience:

Mature

Multithreaded

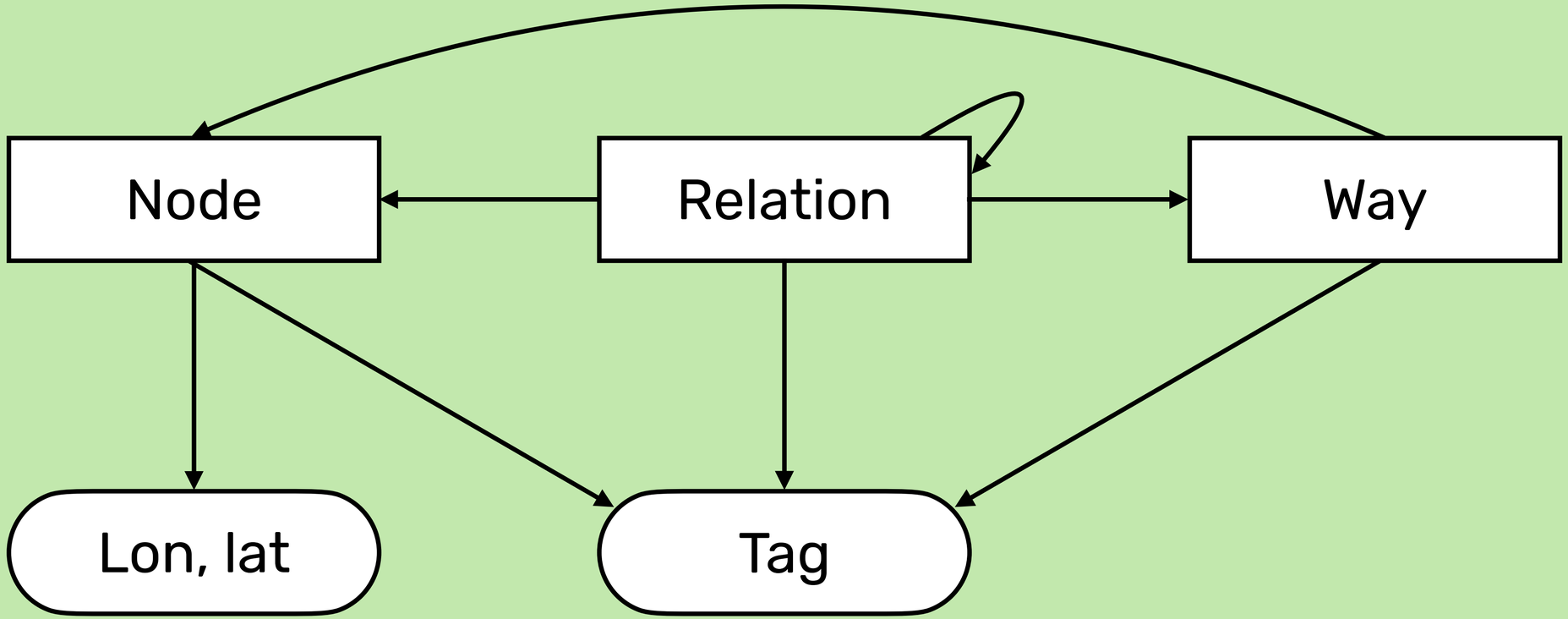
Slightly faster

Mature

Multithreaded

Slightly more convenient

OPENSTREETMAP DATA MODEL



OPEN OPENSTREETMAP FILE

```
let reader = osmpbf::reader::ElementReader::from_path(path)?;

reader.for_each(|obj| {
    match obj {
        Element::Node(node) => ...,
        Element::DenseNode(dense_node) => ...,
        Element::Way(way) => ...,
        Element::Relation(relation) => ...,
    }
});
```

FIND PLACES WHERE PIRATES COULD HIDE

```
if way
    .tags()
    .any(|t| t.0 == "name" && t.1.to_lowercase().contains("pirate"))
{
    result.push(ParsedObject {
        kind: Kind::PirateDen,
        properties: convert_tags_to_properties(way.tags()),
        geometry: Location::Refs(way.refs().collect()),
    });
}
```

FIND ICE CREAM PARLORS

```
if way
    .tags()
    .any(|t| t.0 == "amenity" && t.1 == "ice_cream")
{
    result.push(ParsedObject {
        kind: Kind::IceCreamParlor,
        properties: convert_tags_to_properties(way.tags()),
        geometry: Location::Refs(way.refs().collect()),
    });
}
```

FIND CITIES, TOWNS, AND VILLAGES

```
if node
    .tags()
    .any(|t| t.0 == "place" &&
        (t.1 == "city" || t.1 == "town" || t.1 == "village"))
{
    result.push(ParsedObject {
        kind: Kind::Place,
        properties: convert_tags_to_properties(node.tags()),
        geometry: Location::LonLat(node.lon(), node.lat()),
    });
}
```

CONVERTING TO SOMETHING USABLE



geo

Data types

Algorithms/operations

I/O

Very mature

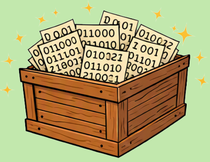
Part of GeoRust:

https://georust.org

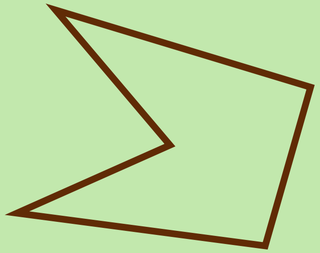
CONVERTING TO SOMETHING USABLE



geo-types



geojson



Way

➡ Polygon



Node

➡ Point

```
"properties": {
    "name": "Rust",
    "name:gsw": "Ruescht",
    "place": "village",
    "population": "3582",
    "wikidata": "Q632924"
}
```

Tag ➡ serde_json::Map

RESULTS

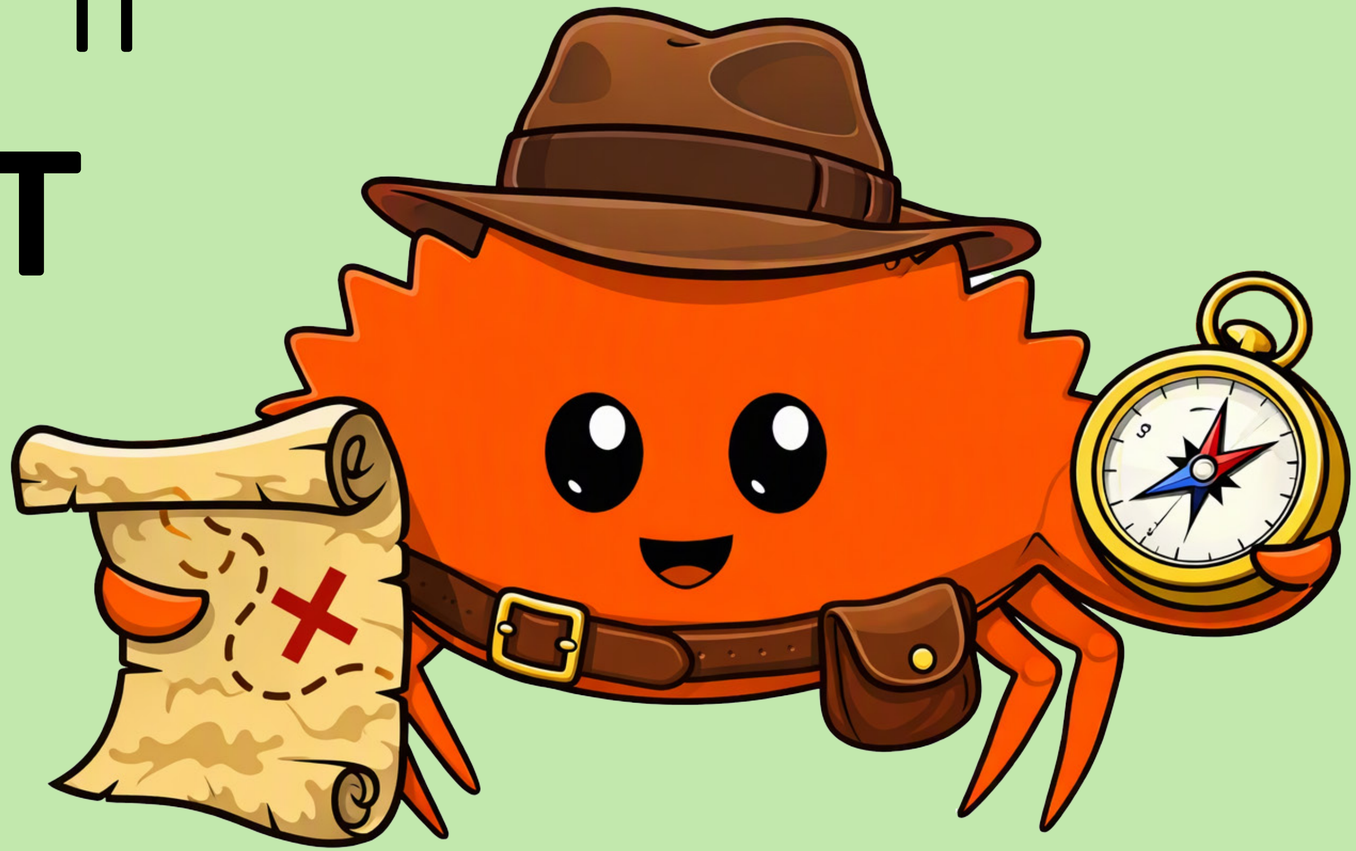
Data preparation took **10m 44s**
on a fast desktop computer

Full code
on GitHub!



Chapter II

THE HUNT



PREVIOUSLY ...

Question:

“Find a place with a pirate den that is very close to an ice cream parlor”

➡ **Spatial query**



SPATIAL INDEXES



kiddo

k-d tree, points



rstar

r*tree, arbitrary geometries



geo-index

k-d tree & r-tree



geohash

linear quadtree



h3

discrete global grid system

SPATIAL INDEXES



kiddo

k-d tree, points



rstar

r*tree, arbitrary geometries



geo-index

k-d tree & r-tree



geohash

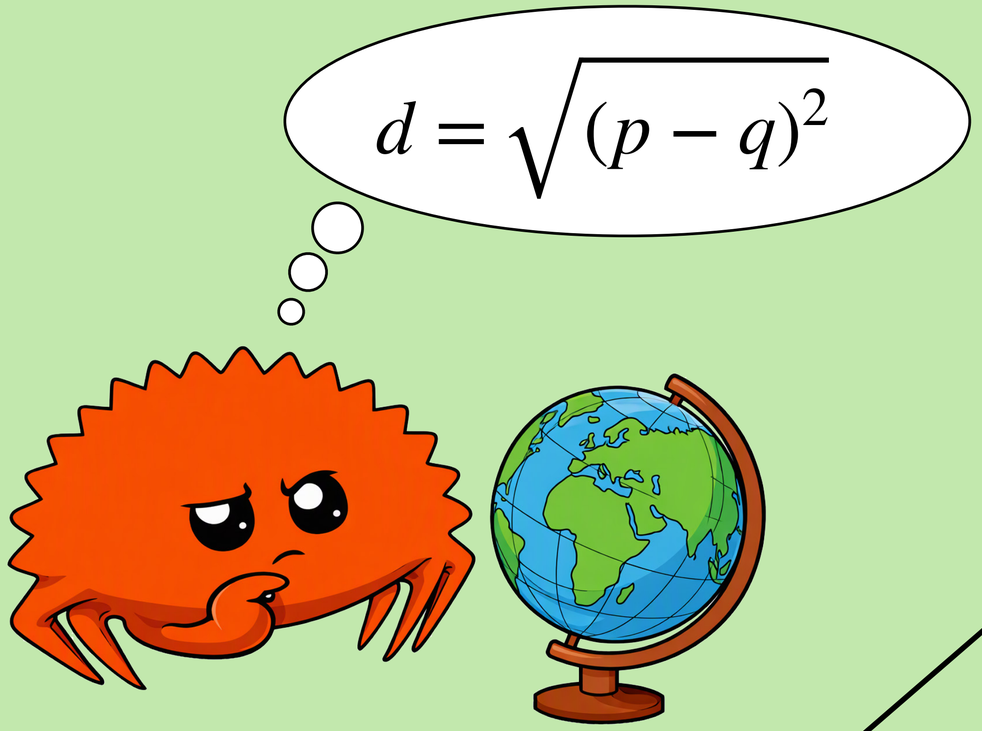
linear quadtree



h3

discrete global grid system

R*TREE NEEDS EUCLIDEAN METRIC




```
"geometry": {
  "type": "Polygon",
  "coordinates": [[
    [ 7.7263213, 48.2659347 ],
    ...
  ]]
}
```

Lon/lat on Earth's surface ➡ **Not Euclidean**

COORDINATE TRANSFORMATION



proj

7.7263213, 48.2659347 → 405479.50, 5346642.51

Longitude, Latitude

WGS84
EPSG:4326

Meters

ETRS89 / UTM zone 32N
EPSG:25832

COORDINATE TRANSFORMATION

```
let from = "EPSG:4326";
let to = "EPSG:25832";
let wgs84_to_utm32n = Proj::new_known_crs(from, to, None)?;
pirate_dens
    .iter_mut()
    .try_for_each(|p| p.transform(&wgs84_to_utm32n))?;
ice_cream_parlors
    .iter_mut()
    .try_for_each(|p| p.transform(&wgs84_to_utm32n))?;
places
    .iter_mut()
    .try_for_each(|p| p.0.transform(&wgs84_to_utm32n))?;
```

OUR QUESTION

“Find a place with
a pirate den that is
very close to an ice cream parlor”

OUR QUESTION

“Find a place with

a pirate den that is

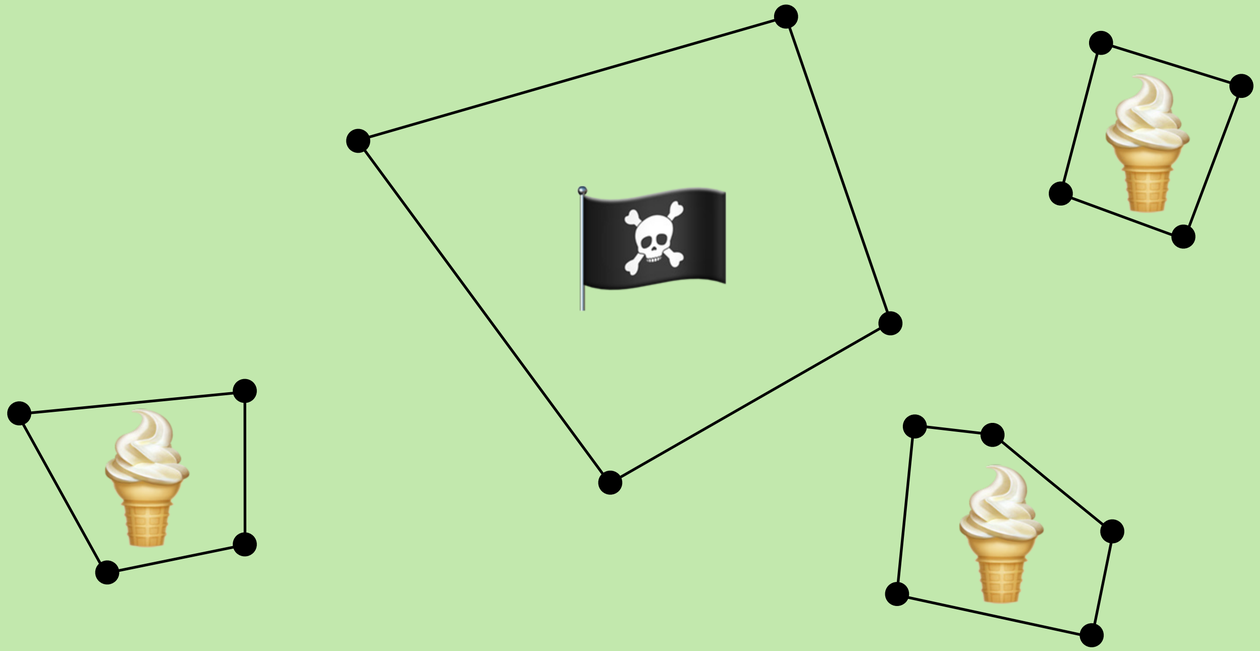
very close to an ice cream parlor”

OUR QUESTION

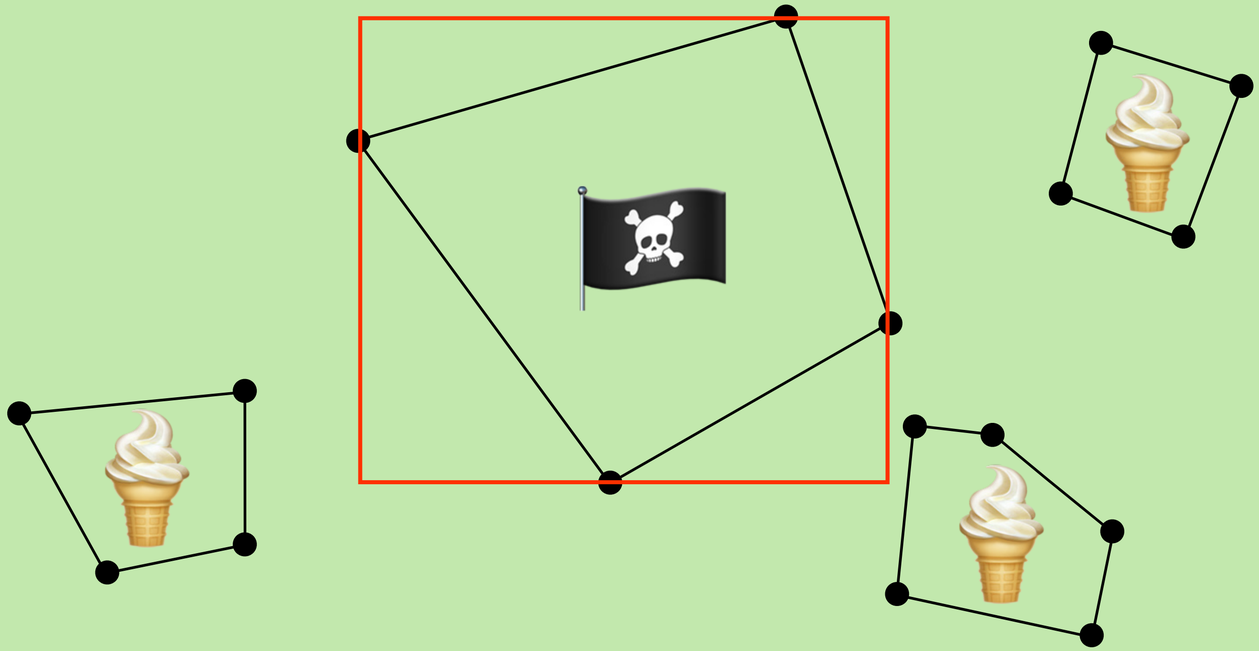
“Find a place with
a pirate den that is
very close to an ice cream parlor”

as close as possible

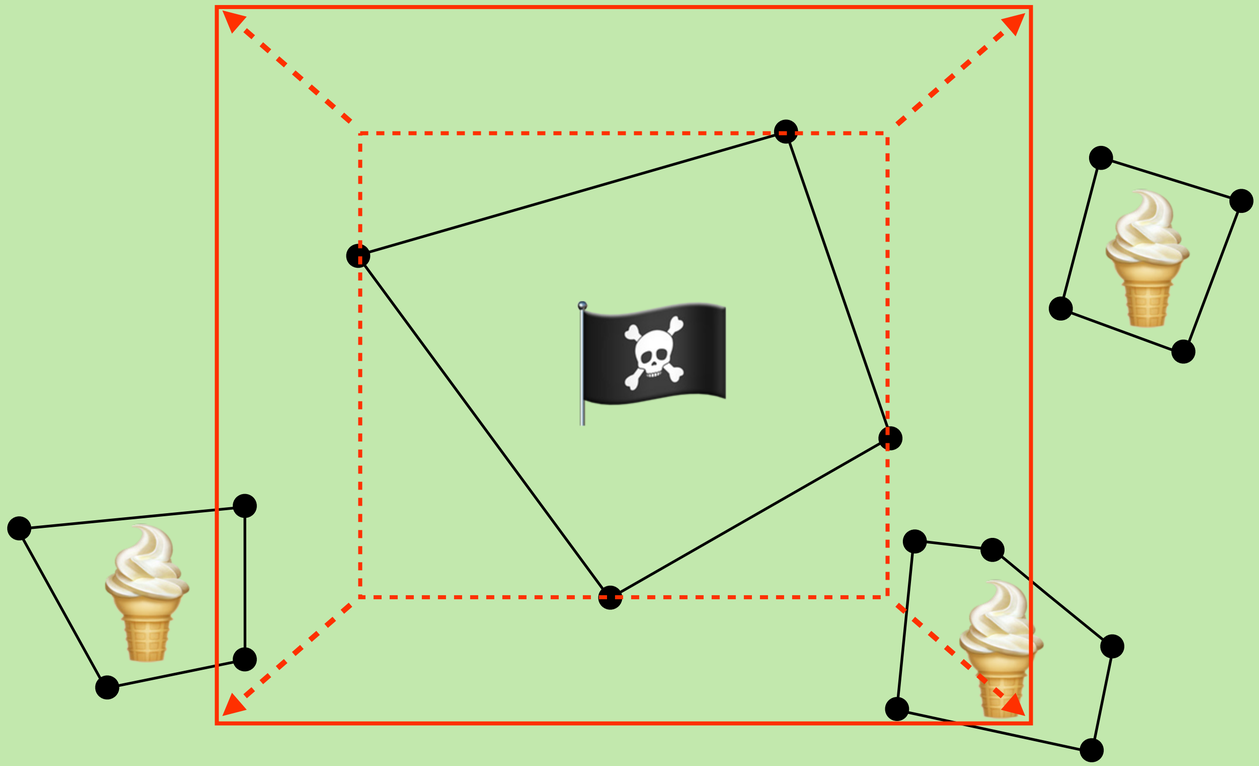
A PIRATE DEN CLOSE TO AN ICE CREAM PARLOR



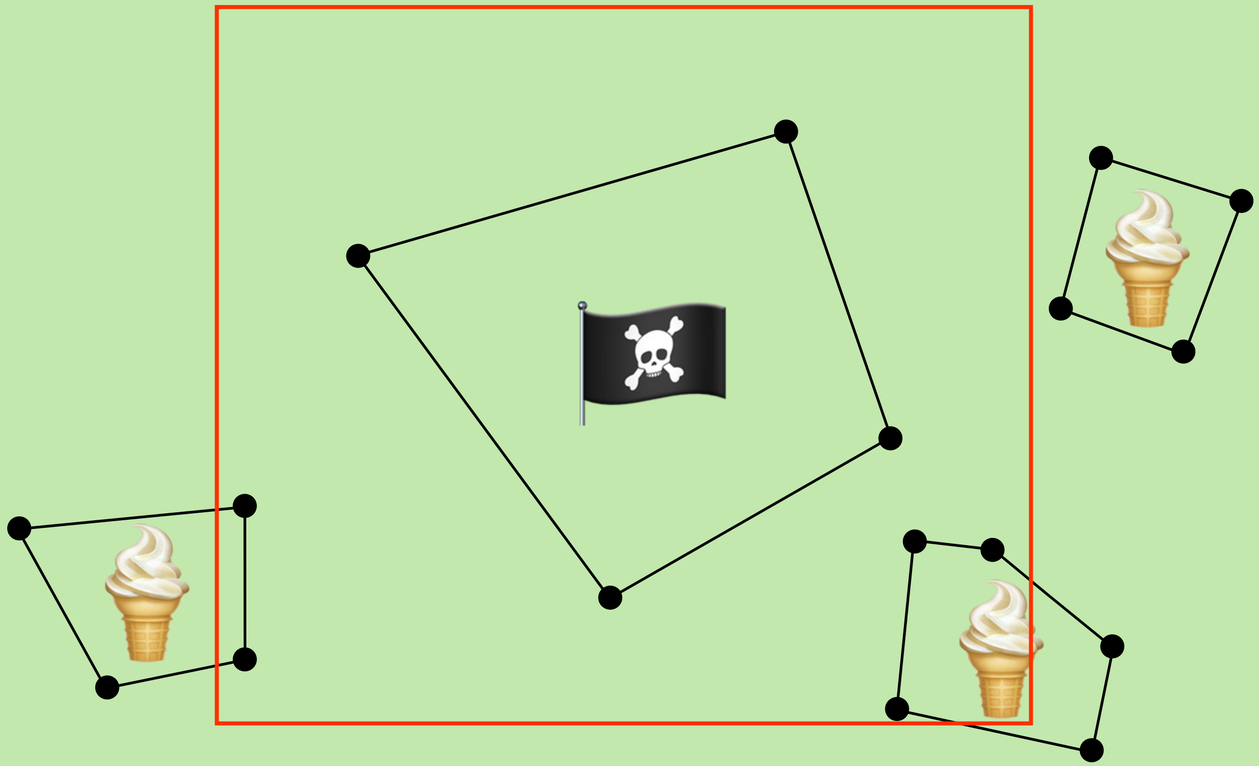
A PIRATE DEN CLOSE TO AN ICE CREAM PARLOR



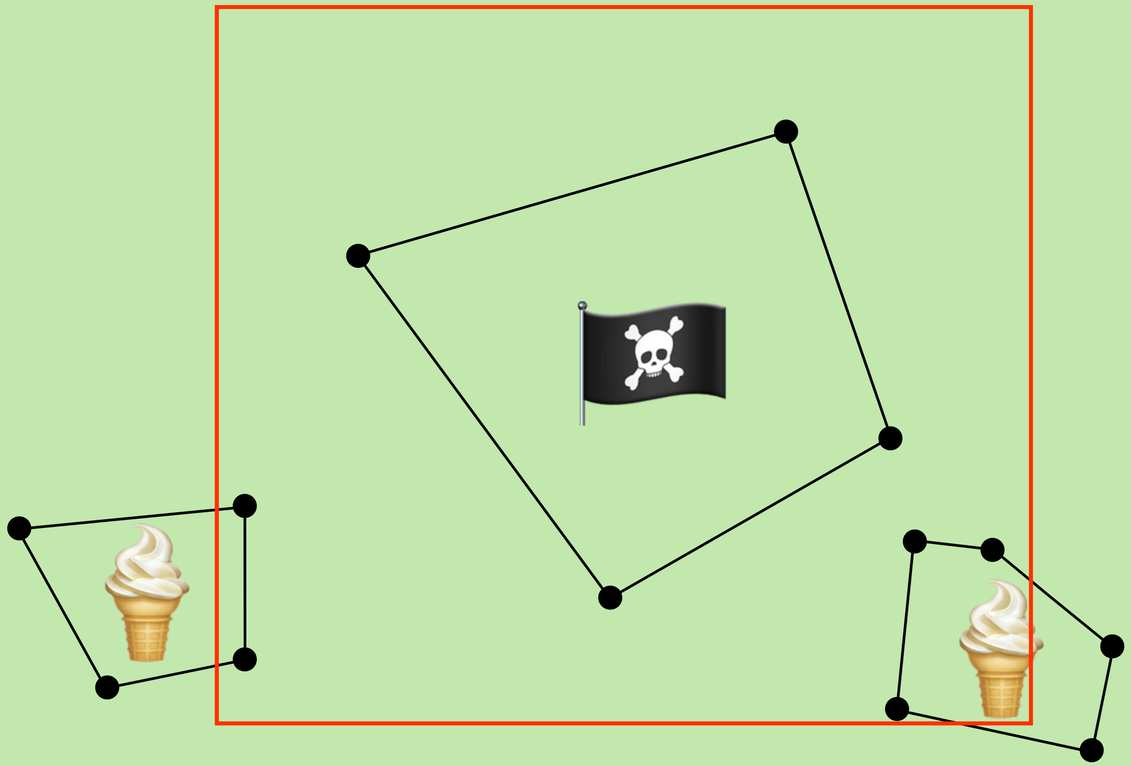
A PIRATE DEN CLOSE TO AN ICE CREAM PARLOR



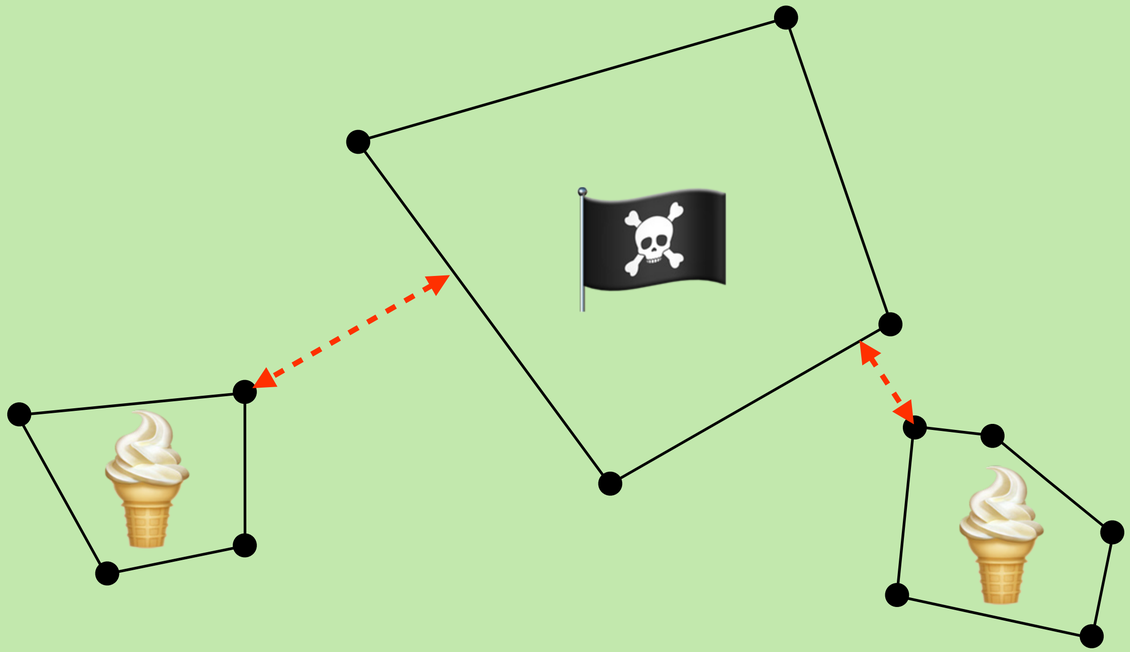
A PIRATE DEN CLOSE TO AN ICE CREAM PARLOR



A PIRATE DEN CLOSE TO AN ICE CREAM PARLOR



A PIRATE DEN CLOSE TO AN ICE CREAM PARLOR



A PIRATE DEN CLOSE TO AN ICE CREAM PARLOR

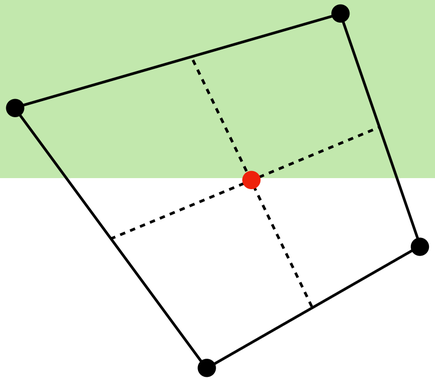
```
let index: RTree<Polygon> = RTree::bulk_load(ice_cream_parlors);

let mut min_pirate_den = (f64::MAX, None);
for p: &Polygon in &pirate_dens {
    let env = p.envelope().expand(100.0);
    for candidate in index.locate_in_envelope_intersecting(&env) {
        let d = Euclidean.distance(p, candidate);
        if d ⩽ 100.0 && d < min_pirate_den.0 {
            min_pirate_den = (d, Some(p));
        }
    }
}
```

OUR QUESTION

“Find a place with
a pirate den that is
very close to an ice cream parlor”

NEAREST **CITY, TOWN, OR VILLAGE**

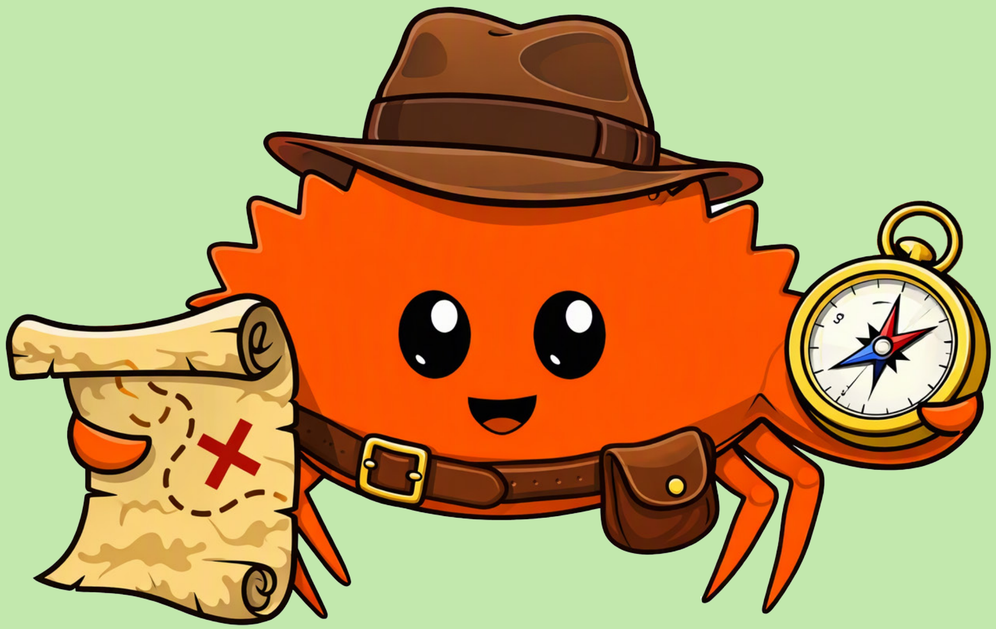


```rust
let index: RTree<Point> = RTree::bulk_load(places);

let pirate_den_centroid = min_pirate_den
    .1
    .context("no pirate den within 100m of an ice cream parlor")?
    .centroid()
    .unwrap();

let nearest_place = index
    .nearest_neighbor(&pirate_den_centroid)
    .context("no nearest place found")?;
```

DEMO TIME!





Epilogue



EVEN MORE **GEOSPATIAL OPERATIONS**

- Spatial queries ✔
- Coord. transformations ✔
- Boolean and buffer
- AI-based image analysis
- Voronoi/Delaunay
- Simplification/generalization
- Routing
- Convex hull
- Map algebra

And many more ...

EVEN MORE **CODE**



geo

Pure Rust

Very mature



geos

C library bindings

Battle-tested

PostgreSQL/PostGIS

➡ **Use both if necessary**

MICHEL KRÄMER

michel.kraemer@igd.fraunhofer.de

github.com/michel-kraemer

s.fhg.de/rust-on-a-map

Full code
on GitHub!



Fraunhofer
IGD