Focus-Glue-Context Fisheye Transformations for Spatial Visualization

by Thanh Cuong Nguyen, Michael Lydeamore, and Dianne Cook

Abstract Fisheye views magnify local detail while preserving context, yet projection-aware, scriptable tools for R spatial analysis remain limited. mapycusmaximus introduces a Focus–Glue–Context (FGC) fisheye transform for numeric coordinates and sf geometries. Acting radially around a chosen center, the transform defines a magnified focus (r_in), a smooth transitional glue zone (r_out), and a fixed exterior. Distances expand or compress via a zoom factor and a power-law squeeze, with an optional angular twist that enhances continuity. The method is projection-conscious: lon/lat inputs are reprojected to suitable CRSs (e.g., GDA2020/MGA55), normalized for stable parameter control, and restored afterward. A geometry-safe engine (st_transform_custom) supports all feature types, maintaining ring closure and metadata. The high-level sf_fisheye() integrates with tidyverse, ggplot2, and Shiny, with built-in datasets and tests ensuring reproducibility. By coupling coherent radial warps with tidy, CRS-aware workflows, mapycusmaximus enables spatial exploration that emphasizes local structure without losing global context.

1 Introduction

Maps that reveal fine local structure without losing broader context face a persistent challenge: zooming in hides regional patterns, while small-scale views suppress local detail. Traditional solutions—insets, multi-panel displays, aggressive generalization—break spatial continuity and increase cognitive load. What if we could smoothly magnify a metropolitan core *while keeping it embedded* in its state-level context?

This package implements a Focus–Glue–Context (FGC) fisheye transformation that continuously warps geographic space. The transformation magnifies a chosen focus region, compresses surrounding areas into a transitional glue zone, and maintains stability in the outer context. In contrast to discrete zoom levels or disconnected insets, this approach operates directly on vector geometry coordinates, preserves topology, and supports reproducible, pipeline-oriented cartography within the R sf and ggplot2 ecosystem.

The intellectual lineage of focus+context visualization traces back to Furnas (1986)'s *degree-of-interest* function, which formalized how to prioritize salient regions while retaining global structure. Sarkar and Brown (1992) and Sarkar and Brown (1994) extended this to geometric distortion, demonstrating smooth magnification transitions for graph visualization. Subsequent innovations explored diverse lenses: hyperbolic geometry for hierarchies (Lamping et al., 1995), distortion-view frameworks (Carpendale and Montagnese, 2001), and "magic lens" overlays (Bier et al., 1993). By 2008, Cockburn et al. (2008)'s comprehensive review synthesized two decades of research across overview+detail, zooming, and focus+context paradigms.

In cartography, the need for nonlinear magnification emerged independently. Snyder (1987) developed "magnifying-glass" azimuthal projections with variable radial scales—mathematical foundations still cited today. Harrie et al. (2002) created variable-scale functions for mobile devices where user position appears large-scale against small-scale surroundings. The crucial breakthrough came from Yamamoto et al. (2009) and Yamamoto et al. (2012): their **Focus+Glue+Context model** introduced an intermediate "glue" region that absorbs distortion, preventing the excessively warped roads and boundaries that plagued earlier fisheye maps. This three-zone architecture proved particularly effective for pedestrian navigation and mobile web services.

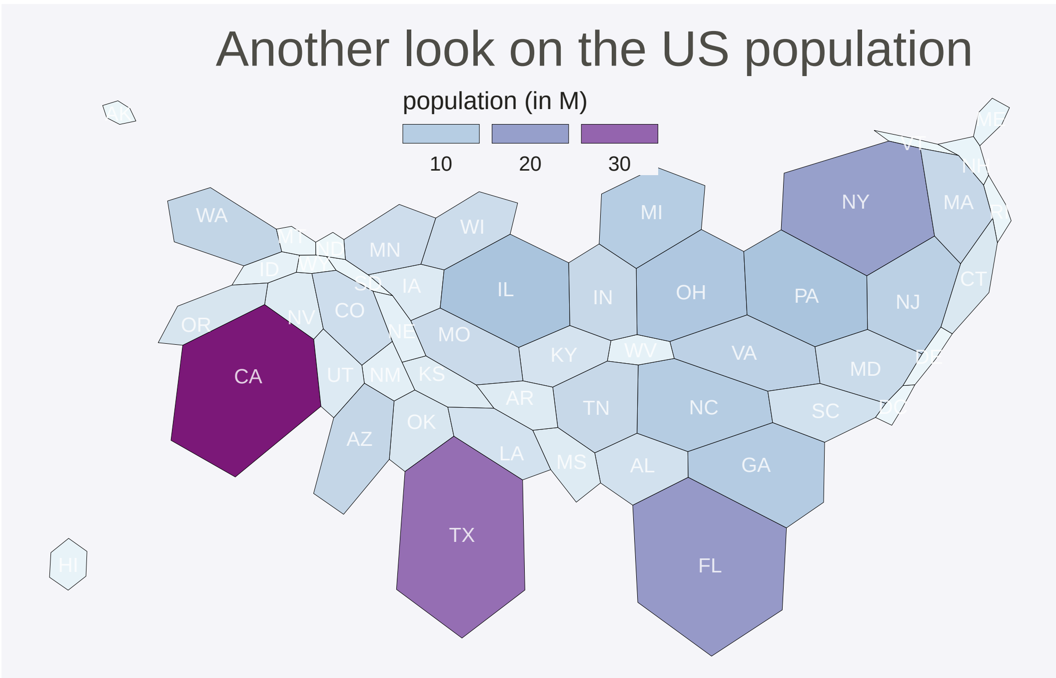
Parallel developments in statistical graphics tackled the "crowding problem"—high-dimensional data collapsing into projection centers. van der Maaten and Hinton (2008)'s t-SNE uses heavy-tailed distributions to spread points, while McInnes et al. (2020)'s UMAP leverages topological methods. Most relevant to our geometric approach: Laa et al. (2020) applies *radial transformations* to tour projections, maintaining the interpretability of linear methods while mitigating overplotting. Implemented in R's tourr package, it demonstrates how well-designed radial warps can reveal structure without the distortions of fully nonlinear embeddings.

Within R's spatial ecosystem, sf (Pebesma, 2018) provides robust vector handling and CRS transformations, while ggplot2 (Wickham, 2016) offers declarative visualization grammar. Yet a gap remained: existing tools addressed *related* distortion needs but not continuous geometric fisheye lenses. This package fills that niche by formalizing an sf-native FGC radial model with controllable zone parameters, optional angular effects, automatic normalization, and safe geometry handling across points, lines, and polygons.

2 Background: Alternative Approaches to the Detail-Context Problem

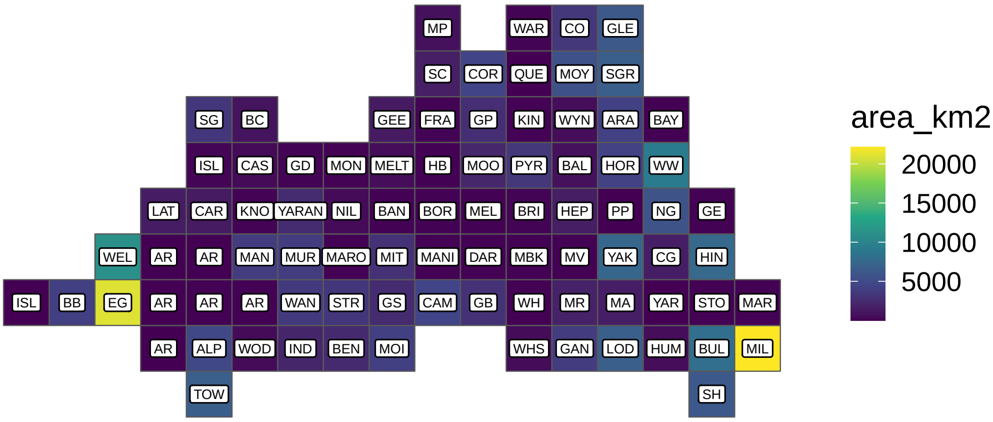
Before examining the mechanics of fisheye transformations, it is important to review how R's spatial ecosystem currently addresses the detail-versus-context tradeoff. This context clarifies why existing solutions, though valuable, do not fully address the need for continuous lens-based warping.

Cartograms: Thematic distortion. The cartogram family (Gastner and Newman, 2004) intentionally distorts geographic areas to encode variables—population density reshapes regions so area becomes proportional to demographic weight.



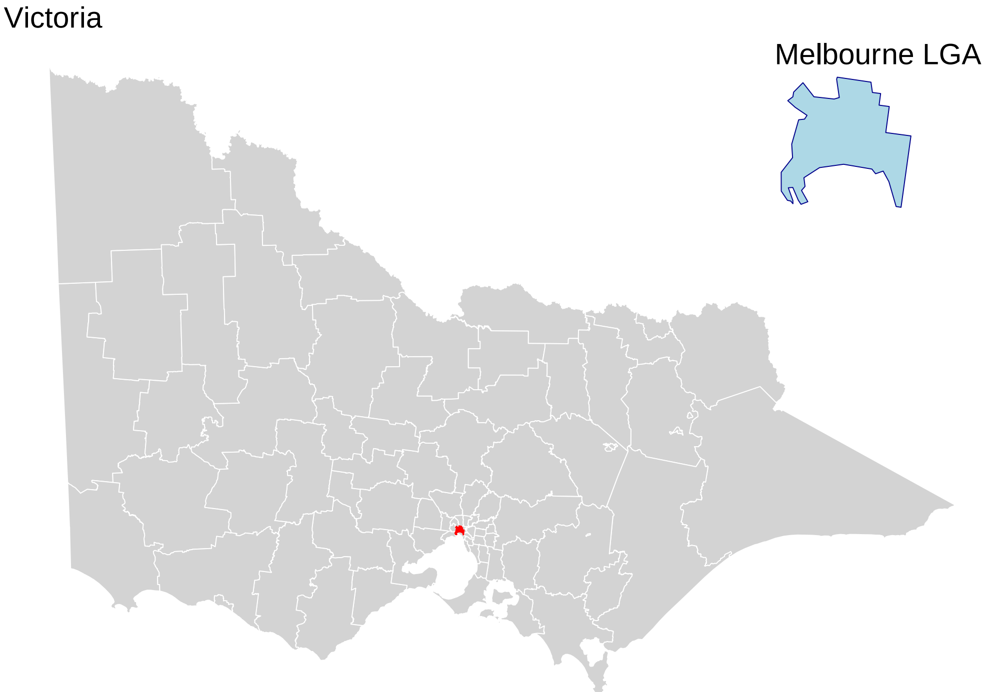
This approach fundamentally differs from focus+context methods. Cartograms substitute spatial accuracy for data encoding, often severely disrupting shapes and adjacencies. For example, a population cartogram enlarges Melbourne while shrinking Mornington, prioritizing thematic insight over geographic fidelity. In contrast, the FGC fisheye transformation preserves relative positions and topology while magnifying a user-selected spatial region rather than a data-driven variable. The use cases are distinct: cartograms address the dominance of a variable in space, whereas fisheye lenses facilitate exploration of local detail within a broader geographic context.

Hexagon tile maps: Discrete abstraction. Packages like `geogrid` and visualizations using `sf::st_make_grid()` replace irregular polygons with regular hexagonal or square tiles, each representing an administrative unit.



As seen in the plot above, tile maps *abstracts away* precise geography entirely, treating space as a topology-preserving tessellation where "neighbors touch" matters more than accurate boundaries. Tile maps excel at avoiding size bias (Mildura gets equal visual weight to Yarra) and creating aesthetic, clutter-free layouts. However, they abandon continuous spatial relationships: you cannot identify precise locations, measure distances, or overlay point data meaningfully. Hexbin aggregation for point data (via `ggplot2::geom_hex()`) serves a different purpose—density estimation—rather than focus+context navigation.

Multi-panel approaches: Spatial separation. Tools like `cowplot::ggdraw()`(Wilke, 2025) create side-by-side views: one panel shows overview, another shows zoomed detail.



These are effective for static reports but require viewers to mentally integrate separate views, and they don't preserve the *embedded* relationship between focus and context within

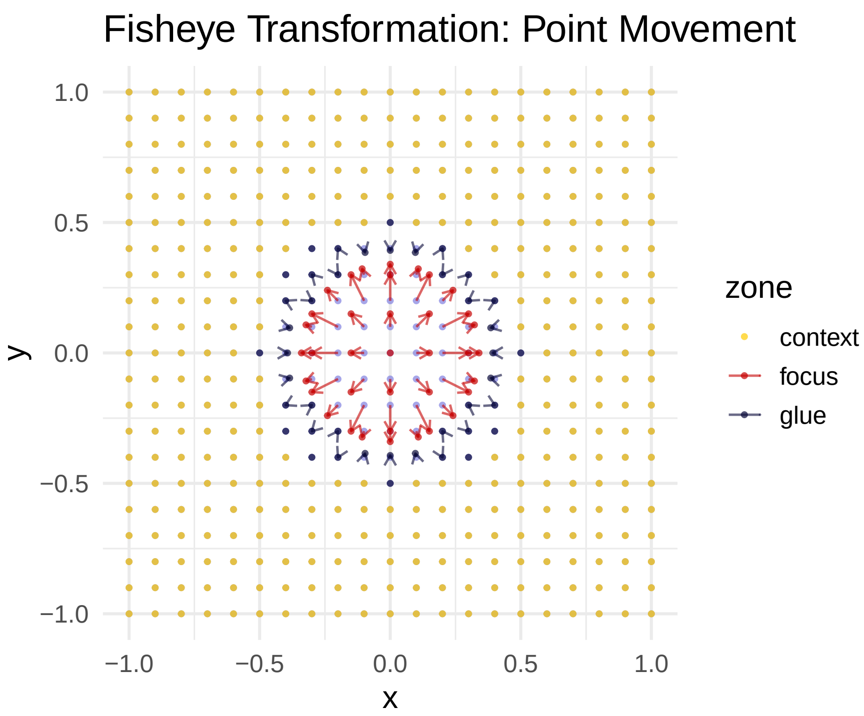


Figure 1: The three zones of an FGC transformation. Points inside the focus (red) expand radially; points in the glue (blue) compress toward the focus boundary; context points (gold) remain fixed.

a single continuous geography. Futhermore, if you introduce one or more elements into the plot like filling value equal to a variable, the audience will have a hard time identify the zoomed detail.

Why FGC fisheye offers something distinct. None of these approaches provide *continuous geometric magnification within a single, topology-preserving map*. Cartograms distort for data, not user-chosen focus. Tile maps abstract away geography. Multi-panel tools spatially separate context. The fisheye lens keeps everything in one frame—roads bend smoothly, metropolitan detail enlarges, but you still see how the city sits within its state. It's a geometric *warp* rather than a data-driven *substitution* or panel-based *separation*. This matters for use cases like: examining hospital networks in Melbourne while maintaining Victorian context, exploring census tracts in a metro core without losing county boundaries, or analyzing transit lines with their regional hinterland visible.

With this landscape established, we now turn to the technical implementation: how does the Focus–Glue–Context transformation actually work, and how does this package make it accessible within R's spatial workflows?

3 Focus–Glue–Context Transformation

Consider a point $P = (x, y)$ in a projected coordinate system. The analyst chooses a center $C = (c_x, c_y)$ and two radii: r_{in} delineating the focus region and r_{out} marking the glue boundary. Points inside the focus magnify, points between the radii focus on the center and then compress according to a smooth curve, and points outside remain unchanged. This radial scheme keeps angular coordinates intact, thereby preserving bearings and relative direction.

3.1 Algorithm

Let (r, θ) denote the polar form of point $P = (x, y)$ relative to center $C = (c_x, c_y)$. The transformation defines a new radius r' via a piecewise function:

$$r' = \begin{cases} \min(z \cdot r, r_{\text{in}}) & \text{if } r \leq r_{\text{in}}, \\ r_{\text{in}} + (r_{\text{out}} - r_{\text{in}}) \cdot h(u;s) & \text{if } r_{\text{in}} < r \leq r_{\text{out}}, \\ r & \text{if } r > r_{\text{out}}, \end{cases}$$

where $z > 1$ is the zoom factor within the focus, $s \in (0,1]$ controls glue compression, and $u = (r - r_{\text{in}})/(r_{\text{out}} - r_{\text{in}})$ normalises the glue radius to $[0,1]$. The function $h(u;s)$ is chosen so that $h(0;s) = 0$, $h(1;s) = 1$, and both the first derivatives and the radii match at the boundaries. We adopt a symmetric power curve:

$$h(u;s) = \begin{cases} \frac{1}{2} \cdot u^{1/s} & \text{if } 0 \leq u \leq 0.5, \\ 1 - \frac{1}{2} \cdot (1-u)^{1/s} & \text{if } 0.5 < u \leq 1, \end{cases}$$

which compresses radii near both boundaries and emphasises the mid-glue region. Analysts seeking outward compression can choose alternative methods (e.g., the `"outward"` mode) that bias the curve towards r_{out}.

The transform optionally introduces rotation within the glue zone to accentuate the flow from detail to context. Let $\phi(u)$ denote the angular adjustment. We employ a bell-shaped profile: $\phi(u) = \rho \cdot 4u(1-u)$, where ρ is the revolution parameter (in radians). This function peaks at the glue midpoint and vanishes at the boundaries, ensuring continuity.

The focus and glue formulas ensure r' and $\frac{\partial r'}{\partial r}$ are continuous at r_{in} and r_{out}. Continuity is essential for maintaining perceptual coherence and avoiding visible creases along the glue boundary. In practice, analysts can tune parameters interactively to obtain the desired amount of magnification and compression, knowing the transform behaves smoothly across the domain.

3.2 Implementation

Spatial datasets vary widely in CRS, extent, feature types, and schemas. mapycusmaximus follows a disciplined staged workflow where each step is explicit, auditable, and invariant to input type. The architecture separates numeric mapping, spatial orchestration, and geometry reconstruction, allowing the core transform to remain small and testable while sf-specific concerns are isolated in thin wrappers.

Workflow and CRS handling

The pipeline proceeds: **sanitize input** → **select working CRS** → **normalize** → **warp** → **denormalize** → **restore original CRS**. Empty geometries are dropped and `sf::st_zm()` enforces 2D coordinates.

The package automatically selects a projected working CRS when operating on geographic data: GDA2020/MGA Zone 55 (EPSG:7855) for Victoria, otherwise UTM inferred from the centroid. This ensures distances are measured in metres and parameters behave consistently. The original CRS is restored on return.

A bounding box defines normalization. With `preserve_aspect = TRUE`, uniform scale $s = \max(s_x, s_y)$ is applied; otherwise axes scale independently. Center resolution occurs before normalization and implements precedence rules via `.resolve_center()`: sf/sfc geometries are reduced to a centroid and transformed to working CRS; numeric pairs with `center_crs` are transformed; numeric pairs without CRS use a lon/lat heuristic; `normalized_center = TRUE` interprets pairs in $[-1,1]$ relative to bbox midpoint. If no center is given, the bbox midpoint serves as default.

Core transformation

At the heart of the package is `fisheye_fgc()`, a vectorized function mapping an $n \times 2$ coordinate matrix to a new $n \times 2$ matrix via the FGC rule. Its contract is minimal: numeric arrays and scalar parameters defining center, radii, magnification, compression, method, and revolution. Internally it converts to polar form, applies the piecewise radial map with smooth boundary conditions, optionally perturbs angle via bell-shaped rotation, and converts back to Cartesian. It attaches diagnostic attributes (zone labels, original and new radii) consumed by plotting utilities but not affecting geometry reconstruction.

Numeric stability at zone boundaries is ensured by clamping expansions in the focus so radii do not exceed r_{in}, and using smooth power curves in the glue so derivatives match across boundaries. The radial mapping is vectorized and runs in linear time in the number of vertices.

Geometry reconstruction

Orchestration is handled by `sf_fisheye()`, which presents the user-facing interface while keeping the numeric core untouched. It validates input, selects working CRS, resolves center, constructs normalization closures, and invokes `st_transform_custom()` to rebuild geometries.

The geometry walker `st_transform_custom()` acts as a drop-in analogue to `sf::st_transform()` but applies an arbitrary coordinate function. For each feature, it extracts coordinates via `sf::st_coordinates()`, yielding a matrix with columns $(x, y, L1, L2, \ldots)$ where L1 and L2 index polygon rings and multi-polygon parts. Geometries are split by type:

- **POINT**: direct warp
- **LINESTRING**: warp each vertex, retain order

- **POLYGON**: process each ring (identified by L1) independently
- **MULTIPOLYGON**: nested by (L1, L2) combinations

After transformation, polygon rings are explicitly closed by forcing first and last vertices to equality: $(x'_1, y'_1) = (x'_n, y'_n)$. This prevents numerical drift when the warp changes ring curvature. Geometries are rebuilt using sf constructors (`st_point()`, `st_linestring()`, `st_polygon()`, `st_multipolygon()`), combined into an sfc with original CRS, and spliced back into an sf if appropriate. Attributes are preserved because only the geometry column is replaced.

Error handling is per-geometry: failures emit a warning, return an empty geometry of the correct type, and continue processing others. This makes the transform robust in batch pipelines without aborting map production on single malformed features.

Table 1 illustrates coordinate transformations across zones for a vertical transect, showing radial expansion in the focus, smooth compression in the glue, and identity mapping in the context.

Design and extensibility

Utilities in `utils.R` provide `create_test_grid()` for diagnostics, `classify_zones()` for labeling, and `plot_fisheye_fgc()` for visualization. Dataset documentation in `data.R` accompanies example layers (vic, vic_fish, conn_fish) used in tests.

The modular architecture enables straightforward extensions. Alternate radial profiles swap `fisheye_fgc()` while retaining the pipeline; additional geometry types extend `st_transform_custom()`; raster integration would follow a similar compute-map-resample pattern. Because modules communicate through simple contracts (matrices in, matrices

Table 1: Coordinate transformation across fisheye zones for selected points on a regular grid

x	y	x_new	y_new	zone	r_orig	r_new
-1.0	-1	-1.000	-1.000	context	1.414	1.414
-0.9	-1	-0.900	-1.000	context	1.345	1.345
-0.8	-1	-0.800	-1.000	context	1.281	1.281
-0.7	-1	-0.108	-0.323	focus	0.316	0.340
-0.6	-1	0.000	-0.340	focus	0.300	0.340
-0.5	-1	0.108	-0.323	focus	0.316	0.340
-0.4	-1	0.000	-0.500	glue	0.500	0.500
-0.3	-1	-0.300	-0.400	glue	0.500	0.500
-0.2	-1	-0.200	-0.400	glue	0.447	0.448

out), evolutions can be undertaken incrementally and verified through the existing test scaffolding.

For multi-layer maps, apply the same parameter set (center, radii, zoom, squeeze, method, revolution) to each layer to maintain spatial alignment. The pipeline ensures coincident warps provided the same working CRS and normalization are used. In practice, analysts capture parameters in a list and pass to `sf_fisheye()` for each layer, simplifying reproducible workflows.

The test suite mirrors the modular structure, covering boundary behaviour, zone labeling, CRS round-trips, ring closure, and performance. Functions follow tidyverse conventions: verb names (`sf_fisheye()`, `plot_fisheye_fgc()`), snake_case parameters, small exported surface. Stability is guaranteed for `fisheye_fgc()`, `sf_fisheye()`, `st_transform_custom()`, and documented utilities.

3.3 Parameters

The principal user interface is `sf_fisheye()`, which accepts an `sf` or `sfc` object and returns an object of the same top-level class whose geometry has been warped in a projection- aware manner. For clarity, we group arguments into data/CRS handling, centre selection, and radial warping, and we make explicit the invariants enforced by the implementation.

Data and CRS. The argument `sf_obj` supplies the features to be transformed. Before any calculation, empty geometries are removed and Z/M dimensions are dropped using `sf::st_zm()`, so that downstream computation operates on a strict $n \times 2$ coordinate matrix. The optional `target_crs` sets the working projected CRS; if provided, the input is transformed via `sf::st_transform()` and the original CRS is restored on return. When `target_crs = NULL` and the input is geographic (lon/lat), a projected working CRS is chosen deterministically from the layer's centroid: for Victoria, Australia, GDA2020 / MGA Zone 55 (EPSG:7855) is used; otherwise a UTM zone is inferred by longitude and hemisphere. This choice ensures the fisheye operates in metric units with bounded distortion across the extent of interest. The `preserve_aspect` flag governs normalisation: with `TRUE` (default) a uniform scale $s = \max(s_x, s_y)$ is applied, where s_x, s_y are bbox half-spans; with `FALSE`, independent scales are used per axis. Uniform scaling preserves circular symmetry of the focus and glue; per-axis scaling yields an elliptical interpretation that can be useful for long, narrow extents but should be used deliberately. Degenerate cases ($s_x = 0$ or $s_y = 0$) are handled by substituting a unit scale to avoid division by zero.

Centre selection. The lens centre may be specified in several forms. The preferred interface is `center`, which takes precedence over legacy `cx`, `cy`. If `center` is a numeric pair and `center_crs` is provided (e.g., `"EPSG:4326"`), the point is transformed into the working CRS. If `center_crs` is omitted, a heuristic interprets pairs that lie within $|\text{lon}| \leq 180$, $|\text{lat}| \leq 90$ as WGS84 and transforms them accordingly; otherwise the values are assumed to be already in working-CRS map units. Any `sf`/`sfc` geometry may be used as `center`; non-point centres are combined and reduced to a centroid and then transformed to the working CRS, which is often convenient when the focal area is a polygon (e.g., a CBD boundary) or a set of points (e.g., incident locations). Finally, when the argument {normalized_center = TRUE}, center is interpreted as a pair in $[-1, 1]$ relative to the bbox midpoint and the chosen normalisation (uniform or per-axis). Normalised centres make parameter sets portable across datasets of different extents and are a natural fit for parameter sweeps in reproducible pipelines. If no centre is supplied, the bbox midpoint is used; this default is stable under reprojection.

Radial warping. The radii r_in and r_out define the focus and glue boundaries in the normalised coordinate space and must satisfy r_out > r_in. The interpretation of these radii depends on preserve_aspect. With uniform scaling, a circle of radius r_{in} in unit space corresponds to a circle of radius $r_{\text{in}}\, s$ in map units; with per-axis scaling, the corresponding shape is an axis-aligned ellipse with semi-axes $r_{\text{in}} s_x$ and $r_{\text{in}} s_y$. Inside the focus, distances from the centre are multiplied by zoom_factor; to prevent overshoot, the implementation clamps r' so that points do not cross the r_{in} boundary. Across the glue, squeeze_factor \in (0,1] controls how strongly intermediate radii compress: smaller values create tighter compression near the boundaries and a more pronounced "shoulder" in the middle of the glue; larger values approach a linear transition. The method selects the family of curves used in the glue. The default "expand" applies a symmetrical power law that expands inward and outward halves of the glue to maintain visual balance around the midpoint; "outward" biases the map towards r_{out}, keeping the outer boundary steadier and pushing more deformation into the inner portion of the glue. The optional revolution parameter adds a bell-shaped angular twist inside the glue of magnitude $\rho\, 4u(1-u)$, where u is the normalised glue radius. This rotation vanishes at both glue boundaries and peaks at the midpoint, preserving continuity. Positive values rotate counter-clockwise, negative values clockwise; values are specified in radians.

Inter-parameter interactions and invariants. The following constraints and behaviours are enforced: $r_{\text{out}} > r_{\text{in}} > 0$; zoom_factor ≥ 1 (values close to one yield gentle focus); squeeze_factor in $(0, 1]$ $(= 1$ approaches linear); and monotonicity of the radial map so that ordering by distance from the centre is preserved. The choice of preserve_aspect affects the physical size of radii and thereby the impact of a given parameter set on different datasets; using uniform scaling with a normalised centre yields the most portable configurations. Twisting via revolution is confined to the glue; it does not change radii and therefore does not affect the classification of points into zones. Because angles are modified only in the glue, bearings inside the focus and in the context are preserved.

Return value and side effects. The function returns an object of the same top-level class as its input (sf or sfc). For sf inputs, non-geometry columns are preserved verbatim; only the geometry column is replaced. The original CRS is restored before return so that downstream plotting and analysis code does not need to change. On malformed geometries, the implementation emits a warning and returns an empty geometry of the appropriate family to preserve row count and indices. For exploratory diagnostics, the low-level fisheye_fgc() returns a coordinate matrix with attributes "zones", "original_radius", and "new_radius"; these can be used to plot scale curves and verify parameter effects prior to applying the transform to complex geometries.

3.4 Common choices

Although the parameter space is continuous, certain regimes recur in practice and can serve as reliable starting points. We describe these regimes and articulate the trade-offs that motivate each choice. The recommendations assume the default preserve_aspect = TRUE; when per-axis scaling is enabled, translate radii to semi-axes using the bbox half-spans.

Balanced metropolitan focus within a state. A common narrative emphasises a city region while retaining a recognisable state outline. Choose r_{in} to enclose the urban footprint (often 0.30–0.35) and r_{out} to provide a broad glue buffer (0.55–0.70). A zoom_factor of 1.5–2.0 provides visible enlargement without overwhelming the transition. Pair this with squeeze_factor = 0.25\text{--0.40}, which gently compresses surroundings while maintaining smoothness. The "expand" method yields a balanced appearance in which the mid-glue region visibly bridges detail and context. If preserving the outer coastline or boundary is paramount (e.g., for policy maps where the edge must remain stable), `"outward"` can be substituted to reduce outer drift at the cost of slightly stronger inner squeeze.

Dense line networks and flows. When the layer of interest is line-heavy (transport corridors, flows, hydrology), kink introduction and overplotting are the primary risks. Reduce glue compression and avoid large twists: `squeeze_factor \ge 0.35` (ideally 0.40–0.60) coupled with `revolution \le 0.2` radians keeps linework legible while still communicating focus. The `"expand"` method is generally preferable because its symmetric treatment of the glue reduces inflections near r_{in} and r_{out}. When in doubt, plot a radius-vs-radius diagnostic from `fisheye_fgc()` to confirm that the derivative remains near one at boundaries.

Polygon-dominated maps and choropleths. For administrative regions, land-use parcels, or other polygon-dense layers, slightly stronger compression in the glue is tolerable because viewers rely on silhouette and adjacency rather than precise edge angles. Settings such as `{squeeze_factor = 0.25\text{ - }0.40}` with `zoom_factor = 1.6\text{ - }2.2}` and either `"expand"` or `"outward"` often work well. We recommend `revolution = 0` for publication unless the swirl is part of the intended rhetoric; twists, while visually engaging, can distract from choropleth encoding and complicate legend interpretation.

Small multiples and parameter sweeps. Analysts frequently compare scenarios across maps (e.g., different thresholds or temporal slices). Portability of parameters is maximised by using a normalised centre (`normalized_center = TRUE`) with `preserve_aspect = TRUE`. This yields consistent radii across datasets of different extent and makes small multiples directly comparable. A pattern that works well is to fix $r_{\text{in}}, r_{\text{out}}$ and `squeeze_factor`, and vary `zoom_factor` over a short range (e.g., $1.3, 1.6, 2.0$). Faceting these outputs produces a transparent narrative of how emphasis changes with magnification.

Choosing radii from map scale. When stakeholders communicate distances in kilometres or miles, convert desired physical radii to unit radii using the bbox half-span. With `{preserve_aspect = TRUE}`, $r_{\text{in}} = d/s$ where d is the intended focus radius in map units (metres for metric projections) and s is the larger half-span of the bbox. This rule allows quick calibration: for a state with half-span 250 km, a desired 75 km focus corresponds to $r_{\text{in}} \approx 0.30$. For per-axis scaling, choose semi-axes independently: $r_{\text{in,x}} = d_x/s_x$, $r_{\text{in,y}} = d_y/s_y$, noting that the current implementation interprets r_{in} as a single scalar and therefore realises an ellipse only through `preserve_aspect = FALSE`.

Centres for reproducibility. To avoid ambiguity in collaborative settings, prefer specifying `center` either as an `sf` geometry (whose CRS is explicit) or as a lon/lat pair with `center_crs = "EPSG:4326"`. Numeric pairs without CRS are accepted but rely on heuristics. When the focal area is itself a polygon or multi-polygon, passing that object as `center` ensures the centroid is derived from the same dataset used for the map, improving reproducibility and intent.

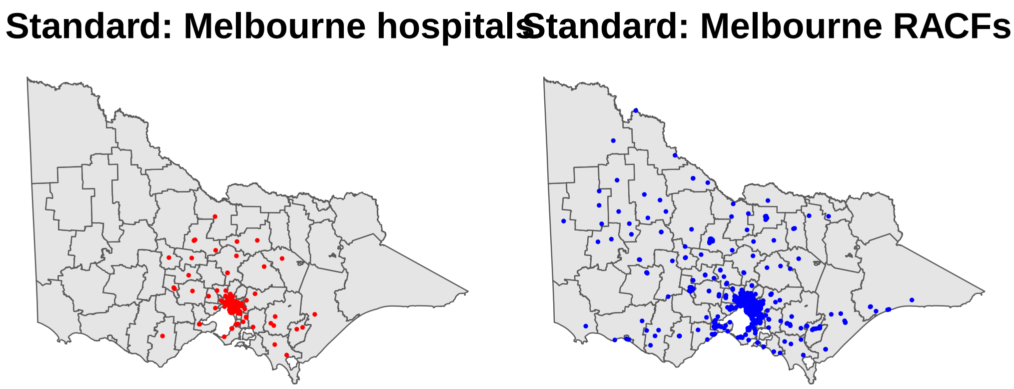
CRS considerations. Leaving `target_crs = NULL` suffices for most lon/lat inputs because the working CRS is chosen deterministically. Projects that maintain a standard grid (e.g., local government dashboards) should specify `target_crs` to improve cross-report comparability. Avoid switching working CRS between layers that will be overlaid; doing so changes the meaning of normalised radii and will misalign warps.

Publication vs. exploration. For exploratory notebooks and talks, small nonzero `revolution` values (≤ 0.3 radians) can help audiences perceive continuity across the glue. For manuscripts and dashboards, prefer `revolution = 0`. Similarly, start with `"expand"` and adopt `"outward"` only when outer stability is an explicit requirement. Always annotate or at least describe the distortion in figure captions so readers do not mistake warped areas for standard projections.

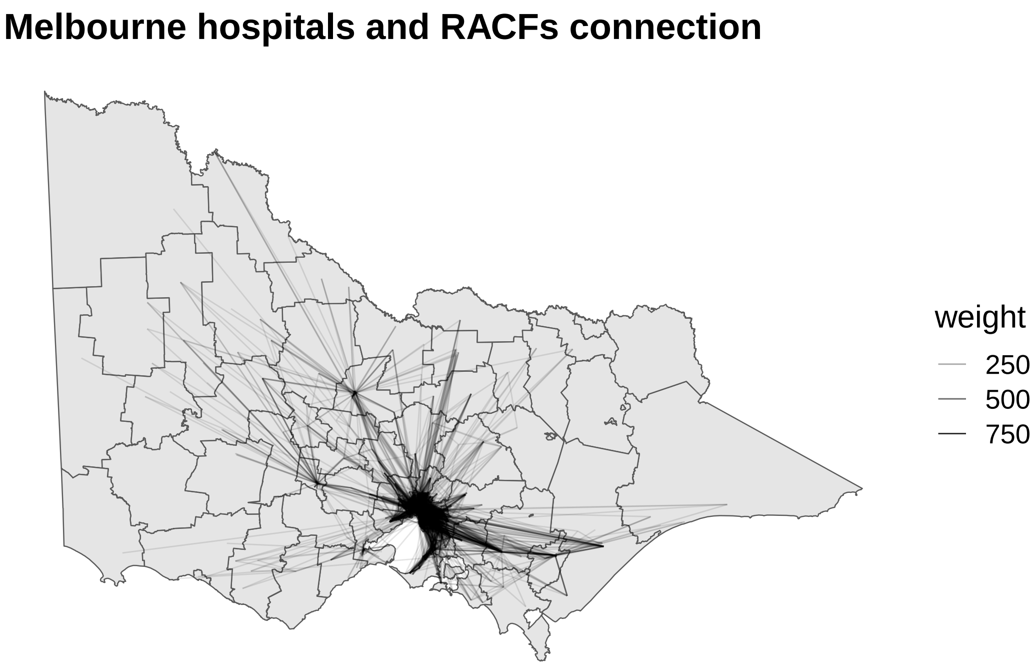
4 Examples of use

We demonstrate mapycusmaximus on Victorian ambulance dispatch data, progressively building complexity to show how the fisheye transformation reveals patterns invisible in standard plots.

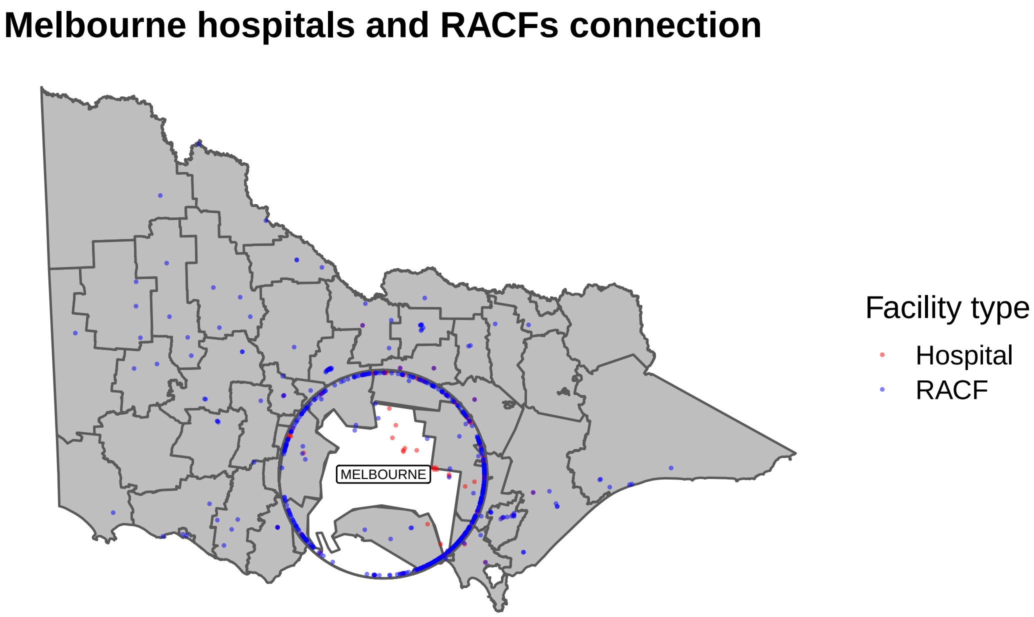
4.1 Hospital locations within state context



First, we can see that in normal approach, we can plot the location of each hospital and each Residential Aged Care Facilities (RACFs) in Melbourne. The map is a bit cluttered, and it is hard to see the relationship between hospitals and RACFs. However, we can still make some sense that the number of RACFs is quite overwhelming comparing to the number of hospitals. The clustering in the metropolitan area of Melbourne is overwhelming and if we put the two plot overlapping each other, then most of the hospitals and RACFs are not visible anymore.

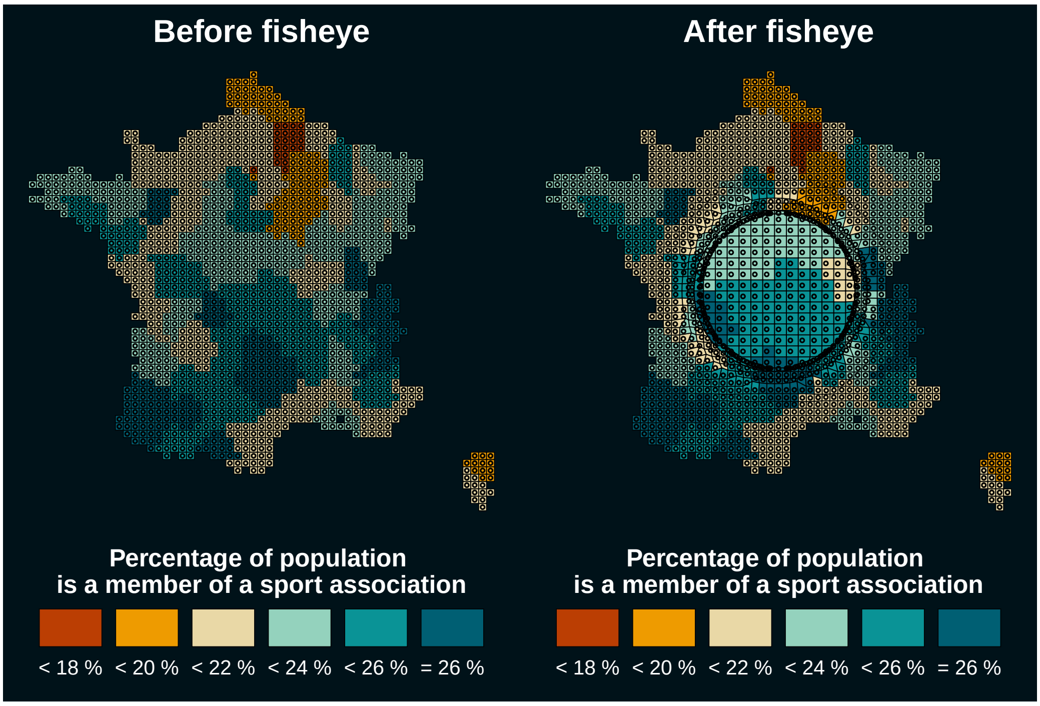


We can go one step futher and displaying the connection or transportation between hospital and RACFs as linestring on the map, with the weight of the line representing the number of ambulance dispatches. The transparency of the line is also proportional to the weight. As we can see know, the central area of Melbourne now complete lost and we can no longer see the spacial structure underneath anymore.



At zoom level at 20, we can directly zoom in and display the Melbourne region to show the number of hospital and RACFs in that LGA, but still reserving the map and the area around with the VIC map to show the spacial structure. Futhermore, since this is just still one `sf` object, multiple calculation that apply to a global context still can be procedd and will not be effected by the fisheye zoom effect.

4.2 France example



5 Discussion

HERE YOU SUMMARISE WHAT THE PAPER CONTRIBUTED IN ONE PARAGRAPH AND SUGGEST NEW WORK THAT MIGHT BE DONE THAT YOU DIDN'T HAVE TIME TO DO

References

E. A. Bier, M. C. Stone, K. Pier, W. Buxton, and T. D. DeRose. Toolglass and magic lenses: The see-through interface. In *Proceedings of SIGGRAPH '93*, pages 73–80, 1993. doi: 10.1145/166117.166126. [p1]

M. S. T. Carpendale and C. Montagnese. A framework for unifying presentation space. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology*, pages 61–70, 2001. doi: 10.1145/502348.502371. [p1]

A. Cockburn, A. Karlson, and B. B. Bederson. A review of overview+detail, zooming, and focus+context interfaces. *ACM Computing Surveys*, 41(1):1–31, 2008. doi: 10.1145/1456650.1456652. [p1]

G. W. Furnas. Generalized fisheye views. In *Proceedings of CHI '86*, pages 16–23, 1986. doi: 10.1145/22627.22342. [p1]

M. T. Gastner and M. E. J. Newman. Diffusion-based method for producing density-equalizing maps. *Proceedings of the National Academy of Sciences*, 101(20):7499–7504, 2004. doi: 10.1073/pnas.0400280101. URL https://www.pnas.org/doi/abs/10.1073/pnas.0400280101. [p2]

L. Harrie, L. T. Sarjakoski, and L. Lehto. A variable-scale map for small-display cartography. In *Joint International Symposium on Geospatial Theory, Processing and Applications*, pages 1–6, 2002. [p1]

U. Laa, D. Cook, and S. Lee. Burning sage: Reversing the curse of dimensionality in the visualization of high-dimensional data, 2020. URL https://arxiv.org/abs/2009.10979. [p2]

J. Lamping, R. Rao, and P. Pirolli. A focus+context technique based on hyperbolic geometry for visualizing large hierarchies. In *Proceedings of CHI '95*, pages 401–408, 1995. doi: 10.1145/223904.223956. [p1]

L. McInnes, J. Healy, and J. Melville. Umap: Uniform manifold approximation and projection for dimension reduction, 2020. URL https://arxiv.org/abs/1802.03426. [p1]

E. Pebesma. Simple features for r: Standardized support for spatial vector data. *The R Journal*, 10:439–446, 2018. ISSN 2073-4859. doi: 10.32614/RJ-2018-009. https://doi.org/10.32614/RJ-2018-009. [p2]

M. Sarkar and M. H. Brown. Graphical fisheye views of graphs. In *Proceedings of CHI '92*, pages 83–91, 1992. doi: 10.1145/142750.142763. [p1]

M. Sarkar and M. H. Brown. Graphical fisheye views. *Communications of the ACM*, 37(12): 73–84, 1994. doi: 10.1145/198366.198384. [p1]

J. P. Snyder. "magnifying-glass" azimuthal map projections. *The American Cartographer*, 14(1): 61–68, 1987. [p1]

L. van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL http://jmlr.org/papers/v9/vandermaaten08a.html. [p1]

H. Wickham. *ggplot2: Elegant Graphics for Data Analysis*. Springer, 2016. doi: 10.1007/978-3-319-24277-4. [p2]

C. O. Wilke. *cowplot: Streamlined Plot Theme and Plot Annotations for 'ggplot2'*, 2025. URL https://wilkelab.org/cowplot/. R package version 1.2.0. [p3]

D. Yamamoto, S. Ozeki, and N. Takahashi. Wired fisheye lens: A motion-based improved fisheye interface for mobile web map services. In A. S. Carswell, James D.and Fotheringham and G. McArdle, editors, *Web and Wireless Geographical Information Systems*, pages 153–170, Berlin, Heidelberg, 2009. Springer Berlin Heidelberg. ISBN 978-3-642-10601-9. doi: https://doi.org/10.1007/978-3-642-10601-9_11. [p1]

D. Yamamoto, S. Ozeki, N. Takahashi, and S. Takahashi. A fusion of multiple focuses on a focus+glue+context map. In *Advances in Cartography and GIScience*, pages 23–37. 2012. doi: 10.1007/978-3-642-29934-6_2. [p1]

Thanh Cuong Nguyen
Monash University
Department of Econometrics and Business Statistics
Melbourne, Australia
https://alex-nguyen-vn.github.io
ORCiD: 0000-0000-0000-0000
thanhcuong10091992@gmail.com

Michael Lydeamore
Monash University
Department of Econometrics and Business Statistics
Melbourne, Australia
https://www.michaellydeamore.com
ORCiD: 0000-0001-6515-827X
michael.lydeamore@monash.edu

Dianne Cook
Monash University
Department of Econometrics and Business Statistics
Melbourne, Australia
https://www.dicook.org
ORCiD: 0000-0002-3813-7155
dicook@monash.edu