

**APPENDIX A**  
**DATA STRUCTURES**

## Regions

```

typedef struct regionlist /* List of all Regions (polygons) */
{
    int     regionNum; /* Region number */
    char    *name; /* Region name */
    double  area, desiredArea; /* Current and desired area */
    double  initialArea; /* Starting area */
    double  data; /* Variable data for cartogram base */
    double  areaError; /* Percent error from desired area */
    double  perimeter; /* Length of perimeter (for Edge Prop. Springs) */
    double  initialPerimeter; /* Initial length of perimeter */
    int     hasFixedPoint; /* Does Region have a fixed point? 0=No, 1=Yes */
    int     numRegionPoints; /* Number of points (vertices) in this region */
    int     *regionPoint; /* Pointer to active array of point indices */
    int     numOrigPoints; /* Number of original points in this region */
    int     *origPoint; /* Array of indices into original Point array */
    int     numSimpPoints; /* Number of simplified points in this region */
    int     *simpPoint; /* Array of indices into simplified Point array */
    int     numKeyPts; /* Number of key points in this region */

    struct regionlist *prev, *next; /* Next and previous Regions */
    struct edgelist *regionEdge; /* Pointer to active list of edges */
    struct edgelist *origEdge; /* List of original edges */
    struct edgelist *simpEdge; /* List of simplified edges */
    struct intersectlist *intersect; /* List of intersections in this region */
} regionList;

```

## Region edges

```

typedef struct edgelist /* List of all Edges within a Region */
{
    int     edgeNum; /* Edge number (within region) */
    double  length; /* Current length of Edge */
    double  initialLength; /* Initial edge length (for edge prop. springs) */
    double  PPlength; /* Proportional length w.r.t. region perimeter */
    Dpoint3d initOrientation; /* Initial orientation (for orientation springs) */
    int     lastZ; /* Z value of last Orientation cross product */

    struct edgelist *prev, *next; /* Next and previous edges within Region */
    struct pointlist *p[2]; /* 2 Points that make up this Edge */
    struct regionlist *edgeRegion; /* Pointer to parent Region */
} edgeList;

```

## Global point array

```

typedef struct pointlist /* List of all Points (vertices) in the system */
{
    int     pointNum; /* Unique Point number */
    Dpoint3d pos; /* Vertex position (x) */
    Dpoint3d vel; /* Vertex velocity (xdot) */
    double  wt; /* Area force smoothing weight factor */
    double  X0, Y0; /* Initial position (for orientation springs) */
    int     fixed; /* Fixed vertex? 0=No, 1=Yes */
    int     Xindex, Yindex; /* Index into sparse matrix JJT */
    int     keyPt; /* Map Key Point? */
    double  forceX, forceY; /* TOTAL FORCE accumulators in X and Y */
} pointList;

```

## Hinge constraints

---

```
typedef struct hingelist /* List of all Hinges connecting adjacent edges */
{
    int    hingeNum;      /* Hinge number */
    double angle;        /* Current hinge angle */
    double initialAngle; /* Initial hinge angle */

    struct hingelist *prev, *next; /* Next and previous Hinge */
    struct edgelist *w[2];        /* 2 Edges that this Hinge connects */
} hingelist;
```

## Global key points and their simplified edges

---

```
typedef struct keyedgelist /* List key point edges */
{
    int    kpt1, kpt2; /* Endpoints (key points) */
    int    regionNum; /* Region number (to prevent duplicate traversals) */

    struct keyedgelist *prev, *next; /* Next and previous key point edges */
    struct simpgedgelist *first, *last; /* List of simplified edges btwn key pts */
} keyEdgeList;

typedef struct simpgedgelist /* List of simplified key edges & their pts */
{
    int    skpt1, skpt2; /* Endpoints (simplified secondary keypoints) */
    int    numOrigPoints; /* # of original points between endpoints */
    int    *origPtNum; /* Array of original point indices */
    double *u, *v; /* U,V offsets of orig point w.r.t. simp line */
    double initialLength; /* Initial length of simplified line */

    struct simpgedgelist *prev, *next; /* Next and Previous simplified edges */
} simpEdgeList;
```

## Intersections

---

```
typedef struct intersectlist /* List of intersecting edges & their forces */
{
    char    intType; /* Type of intersection: A, B, C, or D */
    int    numEdges; /* # of edges involved in intersection */
    edgeList *w[4]; /* Array of 3 or 4 edge pointers that intersect */
    int    numPoints; /* # of points receiving penalty forces */
    pointList **pt; /* Array of pointers to points */
    double *fX, *fY; /* Intersect Penalty Force to apply to each pt */
    int    numIntersectIter; /* # of consecutive iterations of intesection */
    int    applied; /* Forces was applied during this iteration? */

    struct intersectlist *next; /* Next intersection node for this region */
} intersectList;
```

## Sparse matrices

---

```
typedef struct sparsematrix /* Structure for Sparse Matrix */
{
    int    row; /* Row index into "full-size" matrix */
    int    numData; /* Number of data elements in this row */
    double *data; /* Sparse matrix data */
    int    *col; /* Column index into Sparse matrix data */
} sparseMatrix;
```

**APPENDIX B**  
**SAMPLE PARAMETER FILE**

```

DebugToFile          0
CreateDataInputFile  1
CreateReplayFile     1

UseRoomSimplification  1
NumberOfSimplifications  3
SimplificationOffsetPercent  6.0

NumberOfIterations    20000
EndWhenTotalErrorLTPercent  1.0
TimeStep              0.25
MinimumForceOnPoint   50.0
MaximumForceOnPoint   150.0
StagnationLimit       7

FixBoundaryPoints     0

UseHor/VerOrientationSprings  1
OrientSpringCosinePower(Nos)  30
OrientationSpringConst(Kos)  20.0
OrientationMinSpringAngle  30.0
OrientationMaxSpringAngle  5.0

UseEdgeLenPropSprings  1
EdgeLenPropSprConstant(Keps)  0.15

AreaSpringConstant(Ke)  0.25

ConstraintSmoothingDistance  0.0

UseHinges             1
HingeSpringConstant(Kh)  100
HingeMinimumSpringAngle  5.0
HingeSpringCosinePower(Nh)  10

UseEdgeLengthSprings  1
EdgeLenSpringConstant(Kes)  3.0
EdgeSpringCosinePower(Nes)  10
MinWallLengthFactor  0.05

PreventIntersections  1
PenaltyForceContant(Kp)  2.0

PauseDuringRefresh    0
DrawConstraintForces  1
DrawVertices          0
LineDrawingMagnitude  10.0
CreateElementsAtEnd   1
CreateElementsAllIterations  0
RefreshAfterNiterations  25
PrintPointStatsAfterEachIter  1
ReadMapDataFromInputFile  1
<EndOfFile>

```

**APPENDIX C**

**SAMPLE MAP DATA FILE**

1996\_US\_Population  
 NumberofPoints\_1996\_US\_Pop 744  
 0 7840.36 2427.86  
 1 4530.14 4203.43  
 2 4529.57 4156.08  
 3 4559.49 4085.06  
 4 4564.42 4025.34  
 5 3352.86 4027.24  
 .  
 .  
 743 8698.13 3182.03  
 NumberofRooms\_1996\_US\_Pop 52  
 0 4273084.00 16 AL  
 253 252 251 268 267 266 265 264 263 262  
 261 260 259 258 257 254  
 1 2509793.00 17 AR  
 211 210 166 165 159 160 145 144 143 142  
 141 209 208 207 206 205 204  
 2 4428068.00 16 AZ  
 649 150 151 651 650 607 606 605 604 603  
 602 601 645 646 647 648  
 3 31878234.00 47 CA  
 591 590 600 601 602 603 604 605 606 607  
 608 609 610 611 612 613 614 615 616 617  
 618 619 620 621 622 623 624 625 626 627  
 628 629 630 631 632 633 634 635 636 637  
 638 639 640 641 642 643 644  
 4 381611.00 13 CC  
 726 727 728 729 730 732 731 733 735 734  
 736 725 724  
 5 3822676.00 7 CO  
 80 136 149 150 598 82 81  
 6 3274238.00 7 CT  
 490 491 492 475 474 473 472  
 7 543213.00 4 DC  
 408 407 435 434  
 8 724842.00 7 DE  
 436 437 438 439 426 425 424  
 9 14399985.00 52 FL  
 295 296 297 298 299 300 301 302 303 304  
 305 306 307 261 262 263 264 288 289 290  
 291 292 293 294 308 309 310 311 312 313  
 314 315 316 317 318 319 320 321 322 323  
 324 325 326 327 328 329 330 331 332 333  
 334 335  
 10 7353225.00 23 GA  
 294 293 292 291 290 289 288 264 265 266  
 267 268 277 287 286 285 284 283 282 281  
 280 279 278

11 2851792.00 24 IA  
 50 83 84 85 86 87 88 89 90 91  
 77 76 75 74 64 65 66 67 28 29  
 54 53 52 51  
 12 1189251.00 29 ID  
 563 564 565 566 567 568 569 570 571 538  
 539 540 572 573 574 575 576 577 578 579  
 580 581 582 583 584 585 586 587 588  
 13 11846544.00 33 IL  
 49 114 113 112 111 110 109 108 107 106  
 105 104 103 102 101 100 99 98 97 96  
 95 94 93 92 90 89 88 87 86 85  
 84 83 50  
 14 5840528.00 24 IN  
 131 130 129 128 127 126 125 124 123 122  
 107 108 109 110 111 112 113 121 120 119  
 118 117 116 115  
 15 2572150.00 7 KS  
 79 132 133 134 135 136 80  
 16 3883723.00 36 KY  
 336 337 338 339 340 271 270 269 139 138  
 137 101 102 103 104 105 106 107 122 123  
 124 125 126 127 128 129 130 131 115 116  
 117 341 342 343 344 345  
 17 4350579.00 49 LA  
 171 170 169 168 167 166 210 212 213 214  
 215 216 217 218 219 220 221 222 223 224  
 225 226 227 228 229 230 231 232 233 234  
 235 236 237 238 239 240 241 242 243 244  
 245 246 247 248 249 250 174 173 172  
 18 634487.00 10 LI  
 718 719 720 457 458 459 475 721 722 723  
 19 2843993.00 43 MI  
 63 62 61 60 59 717 716 715 714 713  
 712 711 710 709 708 707 706 705 704 703  
 702 701 700 699 698 697 696 695 694 653  
 654 693 692 691 690 689 688 687 686 41  
 40 39 38  
 20 6750357.00 37 MO  
 663 664 665 666 667 668 669 670 671 672  
 673 674 675 676 677 678 679 356 118 119  
 680 681 682 683 684 655 656 657 658 659  
 660 661 662 685 654 653 652  
 21 5710741.00 20 MA  
 726 743 495 494 493 499 490 472 471 476  
 489 488 487 742 741 740 739 738 737 727  
 22 5071604.00 24 MD  
 417 424 425 426 427 428 429 430 431 432  
 433 409 408 434 435 407 406 405 404 422  
 421 420 419 418  
 23 1243316.00 30 ME  
 526 525 486 485 484 524 523 522 521 520  
 519 518 517 516 515 514 513 512 511 510  
 509 508 507 506 505 504 503 502 501 500  
 24 4657758.00 35 MN  
 37 36 35 34 33 32 31 30 29 28  
 27 26 4 3 2 1 9 8 7 25  
 24 23 22 21 20 19 18 17 16 15  
 14 13 12 11 10  
 25 5358692.00 28 MO



**APPENDIX D**  
**LETTERS OF PERMISSION**

## VITA

### CHRISTOPHER JAMES KOCMOUD

*(pronounced Kuts-mode)*

221 Myra Lou Avenue  
Copperas Cove, Texas 76522  
chris@viz.tamu.edu

### EDUCATION

---

Texas A&M University, College of Architecture - College Station, Texas  
*M.S. in Visualization Sciences, December 1997 (www-viz.tamu.edu)*  
“Constructing Continuous Cartograms: A Constraint-Based Approach”  
is available in PDF format at *www-viz.tamu.edu*

Texas A&M University, College of Engineering - College Station, Texas  
*B.S. in Computer Science, August 1992 (www.cs.tamu.edu)*

### EXPERIENCE

---

Nov 93 - Present    *Facility Project Specialist*  
Texas Engineering Experiment Station, Engineering Program at  
Texas A&M University

Jun 92 - Oct 93    *Scientific Instrument Maker II*  
Aerosol Technology Lab, Dept. of Mechanical Engineering at  
Texas A&M University

Jan 90 - Aug 91    *Computer Specialist, GS-7*  
U.S. Army Test and Experimentation Command at Ft. Hood, TX  
(Cooperative Education, 3 work terms)

### PUBLICATIONS

---

Anand, N.K., A.R. McFarland, F.S. Wong, and C.J. Kocmoud. 1993. Optimization of Aerosol Penetration through Transport Lines. Aerosol Technology Laboratory Report 6441/01/31/93/NKA published for the Nuclear Regulatory Commission NUREG/GR-0006 under grant #NRC-04-89-353.