

# MGED Quick Reference Card

(for version 7.x)

## Starting & Stopping MGED

start MGED with default graphical user interface (GUI) `mgcd`  
 run MGED in classic console mode `mgcd -c`  
 open geometry database file creating new if necessary `mgcd file.g`  
 run a single MGED command on database `mgcd -c file.g cmd`  
 quit MGED `exit` or `quit` or `q`

## Files

Geometry database files in MGED are always *automatically saved* to disk after an edit is made. As such, performing a file "Save" operation manually is not necessary and is not provided by MGED.

open a new or existing geometry database `opendb file.g`  
 close any open geometry database `closedb`  
 save a copy of the currently open database `dump newfile.g`  
 export objects from currently open database `keep newfile.g obj ...`  
 check if file contains duplicate object names `dup file.g`  
 combine a geometry database into existing `dbconcat file.g`  
 eliminate unused space from open database `garbage_collect`  
 display version of currently open database `dbversion`  
 upgrade currently open database to the latest `dbupgrade`  
 import data file as a binary object `wdb_binary -i u c obj file`  
 export binary object to a data file `wdb_binary -o u c file obj`

## BRL-CAD File Name Conventions

binary BRL-CAD geometry database files `.g`  
 ascii BRL-CAD geometry database files (deprecated) `.asc`  
 raw binary headerless 3-channel color image data files `.pix`  
 raw binary headerless 1-channel grayscale image files `.bw`  
 extended UNIX 2D/3D color plot format files `.pl`  
 raytrace command saveview shell script (text) files `.rt`

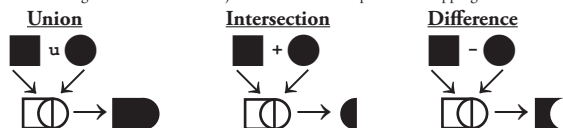
## Geometry Naming Conventions

MGED imposes minimal restrictions on how geometric objects are named. It is up to the individuals and organizations to utilize consistent naming conventions when creating geometry. The below object naming suffix convention is frequently utilized and recommended.

groups / assemblies `no suffix or .g`  
 regions / parts `.r`  
 non-region combinations `.c`  
 primitive solid shapes `.s`

## Constructive Solid Geometry Operations

Constructive Solid Geometry (aka Combinatorial Solid Geometry) is based on three mathematical boolean operations: union, intersection, and difference (aka subtraction). These operators are applied to primitives to form compound objects in MGED using the "u", "+", and "-" notation. Consider the example of combining two primitive object shapes, ■ and ●. The example below shows the resulting CSG combination object when the two shapes are overlapping.



## Getting Help

With non-classic MGED, right-clicking most labels and input fields will provide a description. Additionally, documentation is provided via the Help menu and on-line at <http://brlcad.org/>

obtaining help on all commands `help`  
 obtaining help on a particular command `help command`  
 search for commands that relate to keyword `apropos keyword`  
 display command history for current session `history`  
 record transcript of commands used to file `journal file`  
 list subset of various simulatable GUI actions `press help`

`help`  
`help command`  
`apropos keyword`  
`history`  
`journal file`  
`press help`

## Geometry Information

list the top-level objects `tops`  
 list the objects in currently open database `ls`  
 get a table of contents for current database `t`  
 display the information details for object(s) `l obj ...`

`tops`  
`ls`  
`t`  
`l obj ...`  
`cat obj ...`  
`title`  
`units`  
`tree obj ...`  
`dbfind obj ...`  
`dbfindtree obj ...`  
`paths obj ...`  
`showmats path`  
`get_regions obj ...`  
`eac code ...`  
`summary p r g`  
`idents file obj ...`

get/set title of currently open database `get/set title of currently open database`  
 get/set units of currently open database `get/set units of currently open database`  
 print out CSG hierarchy for object(s) `print out CSG hierarchy for object(s)`  
 display combinations that reference object(s) `display combinations that reference object(s)`  
 display full paths that reference object(s) `display full paths that reference object(s)`  
 list all CSG paths under given object(s) `list all CSG paths under given object(s)`  
 show transformation matrices along a path `show transformation matrices along a path`  
 list all regions referenced by object(s) `list all regions referenced by object(s)`  
 display all regions with given air code(s) `display all regions with given air code(s)`  
 display counts of primitives, regions, groups `display counts of primitives, regions, groups`  
 save region identifier summary to file `save region identifier summary to file`

## Creating Geometry

interactively type in new object parameters `in`  
 create a prototypical primitive object `make type name`  
 create a CSG combination object `comb name op obj ...`

`in`  
`make type name`  
`comb name op obj ...`  
`c name obj op obj ...`  
`r name op obj ...`  
`g name obj1 obj2 ...`  
`build_region prefix #`  
`cp obj objcopy`  
`clone`  
`mv old new`  
`mvall old new`  
`prefix prefix obj`  
`arb rot fallback`  
`cpi cyl cylcopy`  
`make_bb name obj ...`  
`mirror obj new axis`  
`create arb given 3 points, 2 coords of 4th, and thickness 3ptarb`

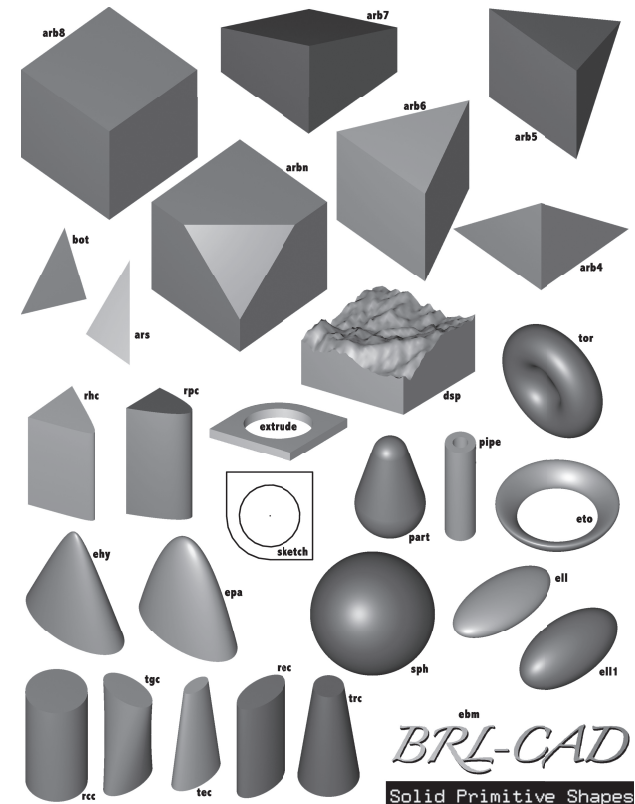
create CSG region (aka "part") combination `create CSG region (aka "part") combination`  
 create group (aka "assembly") combination `create group (aka "assembly") combination`  
 create a region from a range of solids `create a region from a range of solids`  
 create a shallow copy of an object `create a shallow copy of an object`  
 create deep patterned copies of objects `create deep patterned copies of objects`  
 rename an object `rename an object`

rename an object and all references `rename an object and all references`  
 add a prefix to all references to an object `add a prefix to all references to an object`  
 create an arb8 with rotation and fallback `create an arb8 with rotation and fallback`  
 duplicate a cylinder, positioned at end or orig `duplicate a cylinder, positioned at end or orig`  
 make a bounding box around object(s) `make a bounding box around object(s)`  
 mirror an object about the x, y, or z axis `mirror an object about the x, y, or z axis`  
 create arb given 3 points, 2 coords of 4th, and thickness `create arb given 3 points, 2 coords of 4th, and thickness`

## Deleting Geometry

MGED provides *no direct means* to recover deleted geometry, so **delete objects with caution**. Regularly performing geometry database backups (e.g. see the "dump" command) is recommended.

delete object(s) from database `kill obj ...`  
 delete object(s) and all references `killall obj ...`  
 delete object(s), all sub-objects, all references `killtree obj ...`



## Displaying Geometry

display object(s) for editing `e obj ...`

`draw obj ...`  
`d obj ...`  
`erase obj ...`  
`dall obj ...`  
`erase_all obj ...`  
`Z`  
`B obj ...`  
`hide obj ...`  
`unhide obj ...`  
`geomtree`

erase object(s) from the display `erase obj ...`

erase any objects that reference object(s) `dall obj ...`

"zap": clear all objects from the display `Z`  
 "blast": clear all objects & display object(s) `B obj ...`  
 mark object(s) as "hidden" to hide from 'ls' `hide obj ...`  
 unmark object(s) as "hidden" `unhide obj ...`  
 hierarchical geometry browser GUI tool `geomtree`

## Rendering Geometry

raytrace current view to a lingering window `rt -F/dev/X1`  
 raytrace current view to 2048x2048 file `rt -s2048 -o file.pix`  
 raytrace white background hidden-line image `rtedge -W -o file.pix`  
 abort any raytraces started within mged `rtabort`

## Customization

MGED will process a ".mgcdrc" initialization file in your home directory as a sourced Tcl script. This file generally contains defaults set by the GUI but may also include your own customizations including new commands, shortcuts, loadable plugin modules, and custom key bindings.

## Editing Geometry

MGED is a modal editor (akin to "vi") meaning that you have to enter and exit various editing modes. The primary mode states related to editing are VIEWING (default), SOLEDIT, and OBJEDIT. Some commands are only valid in certain modes or change behavior based on mode.

visually illuminate & select combination	<b>ill comb</b>
visually illuminate & select solid primitive	<b>sill prim</b>
enter object-illuminate mode	<b>press oill</b>
get the current editing state	<b>status state</b>
edit a primitive (enter solid edit mode)	<b>sed prim</b>
edit a matrix (enter object edit mode)	<b>oed lpath rpath</b>
add object reference to existing combination	<b>i obj comb</b>
remove object reference(s) from combination	<b>rm comb obj ...</b>
set/get the center of editing transformation	<b>keypoint x y z</b>
manipulate an object's matrix or material	<b>arced path cmd</b>
copy the matrix on one object to another	<b>copymat path1 path2</b>
select matrix path when in pick mode	<b>matpick path1 path2</b>
set a matrix on a given path	<b>putmat path m0 ... m16</b>
apply all matrix transformations down to the primitives	<b>push obj ...</b>
same as push but creates new primitives as needed	<b>xpush obj ...</b>

The geometry editing commands below including the commands related to translation, scaling, and rotation require that MGED be in an edit mode before they can be utilized. The commands implicitly apply to the objects currently selected (e.g. with 'sed' or 'oed') for editing.

set parameter(s) for current edit operation	<b>p val ...</b>
return to viewing mode, accept any edits	<b>accept</b>
return to viewing mode, rejecting any edits	<b>reject</b>
edit selected primitive using a text editor	<b>ted</b>
edit the face of selected arb interactively	<b>facedef face</b>
mirror selected arb face across x, y, or z axis	<b>mirface face axis</b>
permute the vertices of selected arb	<b>permute 8vertices</b>

### TRANSLATING OR MOVING GEOMETRY

move object being edited to relative position	<b>tra dx dy dz</b>
move object being edited to absolute position	<b>translate x y z</b>

### SCALING OR RESIZING GEOMETRY

scale primitive being edited	<b>sca factor</b>
scale combination object being edited	<b>oscale factor</b>
extrude arb face by some absolute distance	<b>extrude face dist</b>

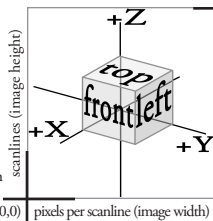
### ROTATING GEOMETRY

rotate primitive being edited	<b>rot x y z</b>
rotate combination object being edited	<b>orot x y z</b>
rotate angle degrees about an arbitrary axis	<b>arot x y z angle</b>
incrementally rotate combination object	<b>rotobj -i dx dy dz</b>
rotate combination about vector	<b>qrot x y z dx dy dz angle</b>
use provided planar coefficients when rotating arb face	<b>eqn A B C</b>

## BRL-CAD Coordinate Systems

BRL-CAD uses a *right-hand* 3D Cartesian coordinate system with real number addressing where "up" is in the positive z-axis (+Z) direction, "left" and "right" are perpendicular to the y-axis, and "front" is towards the positive x-axis (+X) direction.

BRL-CAD uses a *first-quadrant* 2D Cartesian coordinate system with integer addressing where (0,0) is the lower-left corner pixel and (width-1,height-1) is the top-right pixel in an image.



## Text File & Table Editing

Several commands in MGED utilize an external text editor, determined from your environment EDITOR setting, to edit object values. Depending on your shell, you may need to set your EDITOR environment variable before invoking MGED. Bash example: `export EDITOR=pico`

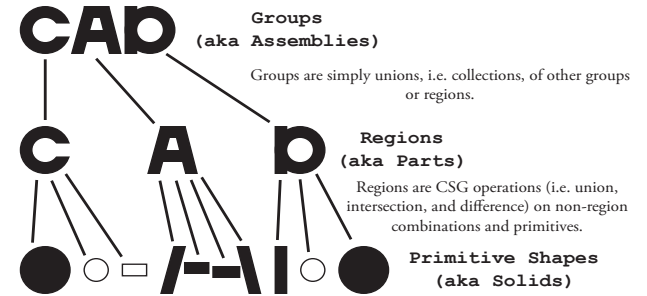
edit a combination using a text editor	<b>red comb ...</b>
edit the region identifier codes for object(s)	<b>edcodes comb ...</b>
edit the combination/region materials	<b>edmater comb ...</b>
print the color table	<b>prcolor</b>
edit the color table codes	<b>edcolor</b>
read/import region identifier codes from file	<b>rcodes file</b>
write region identifier codes to file	<b>wcodes file obj ...</b>
read combination materials from file	<b>rmater file</b>
write combination materials to file	<b>wmater file obj ...</b>
write report of primitive solids to file	<b>solids file obj ...</b>

## Manipulating the View

get/set the various view parameters	<b>view</b>
automatically resize/recenter the view	<b>autoview</b>
redraw the current view	<b>refresh</b>
set the azimuth, elevation, and twist	<b>ae az el tw</b>
set/get the view center	<b>center x y z</b>
set/get the eye point	<b>eye_pt x y z</b>
set/get the viewing direction	<b>lookat x y z</b>
set/get the view size	<b>size size</b>
zoom the view by specified scale factor	<b>zoom scale</b>
set the perspective viewing angle	<b>set perspective angle</b>
translate/move the view relative to current	<b>tra dx dy dz</b>
scale the view size by given factor	<b>sca factor</b>
rotate the view by x, y, z degrees	<b>rot x y z</b>
rotate view about a specified model vector	<b>mrot x y z</b>
rotate viewpoint by specified degrees	<b>vrot xdeg ydeg zdeg</b>
set view using direction and twist angle	<b>qvrot dx dy dz angle</b>
set view using x, y, z angles in degrees	<b>setview xdg ydg zdg</b>
pan the view	<b>sv x y</b>
set the view orientation from quaternion	<b>orientation quat</b>
emulate a knob twist	<b>knob params</b>
control the angle/distance cursor	<b>adc</b>
save the current view orientation to a file	<b>saveview file.rt</b>
load a saved view orientation from a file	<b>loadview file.rt</b>
save current wireframe to a Postscript file	<b>ps file.ps</b>
save current wireframe to a UNIX plot file	<b>plot file.pl</b>
overlay a UNIX plot file onto the display	<b>overlay file.pl</b>

## Analyzing Geometry

analyze the faces of an ARB	<b>analyze arbname</b>
rough estimate of presented area	<b>area</b>
trace single ray from current view or x, y, z	<b>nirt x y z</b>
trace single ray from x, y position	<b>vnirt x y</b>
get/set query_ray behavior settings	<b>gray</b>
check for overlaps (aka interferences)	<b>rtcheck</b>
compute view-dependent surface areas	<b>rtarea</b>
get/set MGED calculation tolerances	<b>tol</b>



## Attributes

In BRL-CAD geometry database files, "attributes" may be used to store arbitrary information, i.e. metadata, on an object. Attributes may be applied to *any* object in the database.

display current attributes for object(s)	<b>attr show obj ...</b>
set the specified attribute on an object	<b>attr set obj atr val</b>
append the specified attribute value	<b>attr append obj a v</b>
modify an object attribute(s)	<b>adjust obj atr nval</b>
delete an object attribute	<b>attr rm obj atr</b>
interactively set visual material properties	<b>mater comb</b>
set object color (red, green, and blue values)	<b>comb_color obj R G B</b>
get region identifier code for specified region	<b>whatid region</b>
list all regions using particular shader(s)	<b>which_shader shdr ...</b>
identify regions with specified air code(s)	<b>whichair code ...</b>
identify regions with specified region id(s)	<b>whichid id ...</b>
incrementally set region id on all regions referenced by object	<b>reid obj #</b>
set material id on all regions referenced by object	<b>remat obj #</b>

## Scripting New Commands in MGED with Tcl

echo, i.e. display or print, the provided text	<b>echo text</b>
pause for the specified amount of time	<b>delay sec usec</b>
get combination CSG structure as a Tcl list	<b>lt object</b>
use shell-style name globbing	<b>set glob_compat_mode 1</b>
use Tcl shell syntax evaluation	<b>set glob_compat_mode 0</b>

Here is an example of writing a custom command called 'get\_primitives' that traverses over all objects in a given combination, printing a list of all primitives encountered. For this example, glob\_compat\_mode is disabled (i.e. set to 0, not the default value of 1) so that there is no need to escape various characters with a preceding "\" slash.

```
set glob_compat_mode 0
proc get_primitives {object} {
  set children [lt $object]
  set prims ""
  if { $children != "" } {
    foreach node $children {
      set name [lindex $node 1]
      set data [db get $name]
      if { [lindex $data 0] != "comb" } {
        set prims [concat $prims $name]
      } else {
        set prims [concat $prims [get_primitives $name]]
      }
    }
  }
  return "$prims"
}
```

Copyright (c) 2006 United States Government  
MGED Quick Reference Card version 4 for BRL-CAD version 7, June 2006  
*designed by Christopher Sean Morrison*

Permission is granted to make and distribute copies of this card provided the copyright notice, the designed by notice, and this permission statement are preserved on all copies.