Bisect 2 angles

This was a little problem we solved for someone to use in....a CAD system - not ours!! It just shows how we helpful we are.

this was undertaken very quickly and only checked very quickly (barely at all)...

the problem is to find the angle which is half the angle between two lines.

firstly some notation to make our work a little easier.

A - the first angle measured in degrees starting at 0 and moving anti-clockwise
B - the second angle measured in degrees starting at 0 and moving anti-clockwise
M - the mid angle measured in degrees starting at 0 and moving anti-clockwise
T - a temp angle measured in degrees starting at 0 and moving anti-clockwise

H - the higher (or greater) of the above angles (A, B)
L - the lower (or lesser) of the above angles (A, B)
why do this? simply to make the calcs simpler later. if we hard code A and B it means we must detect and write all the various cases individually. doing this means we simply put a bit of "sorting out code" at the head....and it all flows on from there.

now some notes:

1. in the above an angle of -20 (clockwise) will be 360 - 20 = 340 (anti-clockwise - which is what we want)

2. although it should be radians, we will use degrees as they are easier to visualise, but please make sure your calculator etc know we are using them rather than radians otherwise the results will be wrong.

3. modulo arithmetic.
the up side of circular work is that you cannot escape, the down side is all this 357, 358, 359, 360/0, 361/1, 362/2 wrapping around stuff. to explain/define this we say it is modulo 360. (recall modulo 2 has 0, 1, 2/0...etc). so when needed we will assume that angles greater than 360 may be reduced to an angle between 0 and 360 by the use of a function that does this. we will use the notation: mod360(Z) which means angle Z must be converted (if necessary) to lie in that region.

P. O. Box 267 Ringwood, Victoria, Australia, 3134
Phone: 61 3 9870 3051
Fax: 61 3 9870 2765
www.cablelabels.com.au
support@cabelabels.com.au

and now....to work.

a) special cases
lets get rid of some silly special cases that just get in the way.

firstly if A = B then its up to you to work out what to do as it does not make sense.

if MOD360(A + 180) = B or MOD360(B + 180) = A then once again its up to you to work out what to do as it does not make much sense.

b) the preamble code

to simplify things do the following:

assume the angles are A < B initially:
H = B
L = A

reverse if required
if A > B then
  H = A
  L = B

(remember A = B has been sorted out above)

we can now safely work with + and - without any problems (modulo underflow)

c) the code

the dirty offset - after you read this the use of H and L should be clear!

assume T firstly (*1*)
T = L

if necessary bounce T around by 180 (*2*)
if (H - L) > 180 then
  T = L + 180

the above is not that clear, so here goes.
we know H > L because we have set it up that way and H <> L because we tested for special cases earlier. so H - L will always give us a posive number. now think about the geometry for a moment, if the difference between the angles measured anti-clockwise is less than 180 degrees we will expect the mid angle to be less than H and greater than L - thats the line marked with (*1*). if the difference is greater than 180 degrees measured anti-clockwise then the mid angle will be either greater than H or less than L - thats the line marked with (*2*).

P. O. Box 267 Ringwood, Victoria, Australia, 3134
Phone: 61 3 9870 3051
Fax: 61 3 9870 2765
www.cablelabels.com.au
support@cabelabels.com.au

now we are ready for the last bit.

M = MOD360(((H - L) / 2) + T)

lets look at this bit.
(H - L) / 2 is simply half the distance between them. if the world was simple the we would just add this to L and the game would be over. but this would only work if (H - L) < 180. now you can see why we used T above to get this first and cater for the other case in advance. the MOD360 is needed to tidy up the mess after all the adds which will take things outside the 0 to 360 range. i've only done it once at the end so be careful using values from intermediate steps as they may be outside the 0 to 360 range.

putting the code together in c.....except for the MOD360 which does not exist in c so make it out of fmod using inbuilt functions.

```
#include <math.h>

#define MOD360(D) fmod(D, 360)

double H, L, T, M, A, B;

if (A == B) {
    /* then its up to you to work out what to do as it does not make sense */
    exit(1);
   }

if ( (MOD360(A + 180) == B) or (MOD360(B + 180) == A) ) {
    /* once again its up to you to work out what to do as it does not make much sense. */
    exit(1);
    }

/* assume the nice condition */
H = B;
L = A;

/* flip it if we are wrong */
if (A > B) {
   H = A;
   L = B;
   }

/* assume simple arrangement */
T = L;

/* and if its messy do it */
if ( (H - L) > 180 )
   T = L + 180;

/* do it */
M = MOD360( ( (H - L) / 2 ) + T );
```


P. O. Box 267 Ringwood, Victoria, Australia, 3134
Phone: 61 3 9870 3051
Fax: 61 3 9870 2765
www.cablelabels.com.au
support@cabelabels.com.au