JET PROPULSION LABORATORY

INTEROFFICE MEMORANDUM
OSASDFM 94-21
28 March 1994

TO: Distribution
From: Hiroshi Kadogawa HK
Subject: Beta Test Version of Code V to COMP Conversion Macro, preliminary macro to convert code V files to COMP "in-files"

Background

Code V is a powerful optical design and analysis program from Optical Research Associates, widely used by optical engineers and designers. The Controlled Optics Modelling Package (COMP) optical analysis program developed by D. Redding at JPL generates high resolution image simulations based on an effective software approach to ray-tracing and diffraction analysis. COMP is also very flexible. Its subroutine form allows direct integration with other code to model the optics portion of complex systems including dynamic, thermal and other effects. Together, (tic, two programs form an effective suite of optical design and analysis software tools.

Introduction

COMP uses an ".in file" 10 describe an optical system. This file can be generated in COMP or independently using a text editor. Code V stores the optical system description in a ".len file" generated in ('ode V. When using both programs for a particular application, I found it redundant to write both a ".len file" for Code V and an "in file" for COMP since the information within the files are essentially the same - it was primarily the format that differed. Moreover. 1 found that most subject optical systems were either already in Code V, or 1 simply preferred generating optical systems in Code V rather than COMP. Therefore, 1 wrote a Code V macro that automatically converts Code V optical prescriptions to COMP format. This saved time. If 1 had the optical system in Code V it was easy 10 use the macro to generate a starting point for COMP. If the optical prescription was not in Code V I still found it faster to build the optical system in Code V. verify its performance and convert the design to COMP' form using the macro. Although still under development, the macro is quite useful for most optical systems. The chief aim of this memorandum is to generate additional conversion macro exposure to increase Code V/COMP user productively and facilitate continual refinement of the macro via distribution and feedback.

Macro Description

The macro generates a backbone COMP input file for basic reflective / refractive optical systems. The conversion macro eliminates the tedious task of writing out global vertex positions and inverse surface normal directions to 10 significant figures for each optical system surface. (Surface aperture definitions are not currently converted, but since their locations are specified, the task is simplified.)

This macro is currently in a "beta testing" stage, which means that it works for the cases tested so far. The entire macro is included in this report in attachment 1, It currently works for basic reflective and refractive optical systems. A Code V global coordinate surface-to- surface raytrace macro (attachment 2), and a surface position / orientation macro (attachment 3) are also included in this report. These macros allow direct comparison between Code V and COMP raytraces and prescriptions to verify that they agree.

Macro Usage

To use the macro:

1) In Code V:

- Start Code V
- Restore or generate the subject optical system
- Type: "in comp_convert"
- Answer the macro prompts
- The backbone COMP .in file is generated as "compfile.out.(version #)"

2) In COMP:

Refer to the "Controlled Optics Modelling Package User Manual"[1] for questions regarding COMP.

a) For geometric analysis

- Rename "compfile.out.(version #)" generated by the Code V macro to the form "filename.in" used by COMP. You can then use a text editor to modify the .in file as required.
- Correct or remove any conflicting surfaces.[2]
- Acid any aperture definitions to the .in file. [3] Set obs (o I in COMP to see aperture/obscuration effects.
- For off-axis field angles you must do the following:

  i) For an aperture slop defined by a unique surface

  - Change the Slop surface EltType to Exit"]'yIK = 9 (obscuring surface) and describe the slop as shown in reference 3.

  ii) For an aperture slop defined by an optical clear aperture

  . Add a new, separate obscuring surface (EltType = 9) infinitesimally ahead of the limiting clear aperture surface. with the same shape (fElt, eElt, etc.) as the optical element. Define the aperture of the new surface as described in reference 3. Increment nElt as required.

  In both cases, increase the "Aperture" parameter in the .in file to define an oversized grid of rays so that the stop, not the ray bundle, limits off-axis throughput.

- Load the .in file and run geometrical raytraces and spot diagrams to verify the model fidelity.

b) For diffraction analysis

. Once the geometric model is verified you can proceed 10 these steps.
- For diffraction analysis, you must add two (2) return surfaces before the image plane, and increment nElt by two (2). In most cases. first return surface should coincide with the image plane, and the second return surface should be near the exit pupil. See reference 4 for more details.
- Fix the exit pupil using the FEX[5] and ORS commands.
- Run COMP and load the new .in file.
- Use the "propagate" and "intensity" COMP functions to check diffraction modelling.

Notes

- Always check the COMP .in file to verify it correctly models the optical system.
- Use the Code V raytrace macro to compare Code V chief and marginal raytraces with COMP raytraces. Geometric spot diagrams are also a quick way to compare results.
- Look out for common COMP errors such as non-sequential surfaces.
- Once the new .in file traces rays properly, the user may modify system parameters (such as number of grid points. etc.) as required for the particular application.
- The macro will not convert certain surface types (for example, > 10th order aspheres, diffractive, toroidal, non-sequential and segmented aperture type surfaces). An error message flags these surfaces. The conversion macro should incorporate these surface types in future versions.
- Read the macro comments for more information.

Conclusion

The conversion macro was written to facilitate. COMP usage under several projects. Without dedicated funding to write the macro, it was written with a minimalist approach - to accomplish the task at hand and no more. Thus, there are many features I would still like to add to the macro. Users are encouraged to send any useful modifications or additions to the macro to the author of this report. I can serve as a central point of contact for information, suggestions and upgrades. Users are welcome to contact me regarding the macro (not COMP) at:

mail: Hiroshi Kadogawa
Jet Propulsion Laboratory
MS 306-388
4800 Oak Grove Drive
Pasadena, CA **91109**

e-mail: hiro@osaserv.jpl.nasa.gov
phone: (818) 354-70')3
fax: (818)393-9471

References

1. Redding, D., et. al., Controlled Optics Modelling Package User Manual, Release 1,0, 1 June 1992, JPL internal document, J}'}, **D-9816.**

2. Ibid, p. 60.

3. Ibid, p. 34-37,124-128.

.4. Ibid, p. 71-84.

5. Ibid, p. 68.

Attachment 1

Code V to COMP Conversion Macro

comp_convert.seq

```
ver,all no
!*****************************************************************************
!*              comp_convert.seq -  Code V macro (BETA TEST VERSION 0.1B)
!*
!*
!* FUNCTION:     Generates a preliminary COMP .in file.
!*
* USAGE:         Type "in comp_convert" within Code V.
*
* PROMPTS FOR:   global surface number
*                zoom position
*                field position
*                wavelength position
*
* OUTPUT:        compfile.out.(version #)
*
!* NOTES:
!*
!*      1.  The Code V macro will not overwrite an existing "compfile.out.(version #)"
!*          file.  It increments the version # to avoid this.
!*
!*      2.  Change output filename from "compfile.out.(version #)" to "filename.in"
!*          for use in COMP.
!*
!*      3.  REVIEW AND MODIFY THE GENERATED .in FILE AS REQUIRED FOR THE
!*          SPECIFIC APPLICATION.  Code V and COMP have many capabilities
!*          not incorporated in this macro.  This macro is intended to quickly
!*          generate a backbone COMP prescription of optical system components
!*          in global coordinates.  This eliminates the uninspiring task of
!*          entering vertex points and psiElt's for each surface.
!*
!*      4.  Please refer to the "Controlled Optics Modelling Package User
!*          Manual" (JPL D-9816) for the many considerations involved in
!*          generating a proper COMP .in file.
!*
!*      5.  DEFINE AN OBSCURING SURFACE AT THE STOP LOCATION TO ACT AS A PHYSICAL STOP.
!*          The conversion macro does not do this, and it is important for off-axis
!*          field angles.  There are two (2) approaches to this issue:
!*
!*          If the stop is a unique surface, then change the stop surface EltType
!*          to EltType= 9 (an obscuring surface) and define the stop diameter
!*          as described in the COMP User Manual.
!*
!*          If the stop is defined by the clear aperture of an optical element, then it
!*          is generally easiest to model the stop by adding a new, separate obscuring
!*          surface (EltType= 9), located infinitesimally ahead of the limiting clear
!*          aperture surface, with the same shape as the optical element.  Define the
!*          aperture of the new surface in accordance with the limiting clear aperture.
!*
!*          Then, in either case, increase the "Aperture" parameter in the COMP .in
!*          file to define an oversized grid of rays so that the stop, not the ray
!*          bundle, limits off-axis throughput.  Set obs to 1 in COMP to see
!*          aperture/obscuration effects.
!*
!*      6.  MANUALLY ADD APERTURES AND OBSCURING SURFACES (AS DESCRIBED FOR THE
!*          APERTURE STOP IN NOTE 5).
!*
!*      7.  FOR DIFFRACTION PROPAGATION YOU MUST ADD AND OPTIMIZE RETURN SURFACES
!*          AS DESCRIBED IN THE COMP USERS MANUAL UNDER THE PFX AND ORS COMMANDS.
!*          ALSO, MODIFY THE APERTURE SIZE AND NGRIDPTS, AS DICTATED BY PUPIL AND
!*          IMAGE SAMPLING REQUIREMENTS.
!*
!*      8.  This program incorporates common default parameters to reduce
!*          repetitive user interactions.  CHECK THE FOLLOWING DEFAULTS FOR
!*          AGREEMENT WITH THE SUBJECT OPTICAL SYSTEM.
!*
!*              a) DEFAULTS UNDER 'DEFAULT COMP CONSTANTS':
!*
!*                 - Source index of refraction = 1.000.
!*                 - extinction coefficients = 0.
!*                 - flux = 1.000.
!*                 - GridType = 1. (annular)
!*                 - Obscuration = 0.
!*                 - nGridpts = 31.
!*
!*              b) OTHER DEFAULT COMP PARAMETERS:
!*
!*                    STOp surface defined as global origin.
!*                    ChfRayDir taken @CodeV object surface (l,m,n).
!*                    For the purposes of this macro, finite conjugate is defined
!*                       defined as an object distance less than 1.0e+07 units.
!*                    For FINITE conjugate systems:
!*                       ChfRayPos placed half the object distance BEFORE
!*                       the first surface (even for virtual objects).
!*                    For INFINITE conjugate systems:
!*                       ChfRayPos located @100 units ahead of surface 1.
!*                    Aperture = Code V EPD.
!*                    xGrid = (1,0,0)
!*                    yGrid = (0,1,0)
!*
!*      9. COMP surface types converted:
```

```
!*
!*            - reflective flats
!*            - reflective surfaces including conics
!*            - refractive surfaces including conics
!*            - reference surfaces
!*            - focal plane
!*            - 10th  order (0 term) aspheric coefficients.
!*              An error message indicates that higher order coefficients (if any)
!*              must be added.
!*            - Y-toxoid surfaces modelled as an asphere.
!*
!* WRITTEN BY:  Hiroshi Kadogawa
!*              Jet Propulsion Laboratory
!*              Pasadena, CA
!*              e-mail :  hiro@osaserv.jpl.nasa.gov
!*
!* REVISIONS:
!*
!*  Date        By      Modification
!* ------       --      ------------
!* 8/2/93       HK      Created.
!* 2/18/94      HK      Added ChfRayPos for general object location.
!* 4/5/94       HK      Added 10th order aspheric reflector and refractor
!*.........................................................................

! VARIABLE DECLARATIONS
!
chk yes          ! enforce explicit declaration
ICI str ^format1  ^format2a ^format2   ^format3  ^format4  ^format5
lcl str ^format6  ^format7  ^format8   ^format9  ^format10
ICI str ^format11 ^format12 ^format13  ^format14 ^format15
lcl str ^format16 ^format17 ^format18  ^format19 ^format20
lcl str ^format21 ^format22 ^format22a ^format23 ^format24 ^format25
lcl str ^format26 ^format27 ^format28  ^format29 ^format30
] cl num ^a_flat_surf ^an_index_match ^a_vacuum          ! boolean variables
1 cl num ^zero ^infinity
1 cl num ^s ^global_sur ^zoom_pos ^field_pos ^wave_pos
1 cl num ^z_source ^n_source ^ext_coef ^wavelength ^flux ^gridtype ^aperture
lcl num ^obscuration ^ngridpts ^previous_n ^current_n ^conic_const
1 cl num ^l_chief ^m_chief ^n_chief ^x_chief ^y_chief ^z_chief
1 cl num ^x_so ^y_so ^z_so ^x_sl ^y_sl ^z_sl
lcl num ^a_sur ^b_sur ^c_sur ^d_sur
] cl num ^x_sur ^y_sur ^z_sur ^l_sur ^m_sur ^n_sur
1 cl num ^num_sur
1 cl num ^initial_x ^final_x ^lsc_x ^lsc_y
1 cl num ^initial_y ^final_y ^msc_x ^msc_y
1 cl num ^initial_z ^final_z ^nsc_x ^nsc_y
lcl num ^elt_type ^e_elt ^f_elt ^prop_type ^n_ecoord

! CONSTANTS
!
^zero==          0.000000000e+00
^infinity==      1.000000000e+18

! DEFAULT COMP CONSTANTS

^n_source==      1.000000000e+00
^ext_coef==      ^zero
^flux==          1.000000000e+00
^gridtype==      1
^obscuration==   ^zero
^ngridpts==      31
^prop_type==     1
^n_ecoord==      6
!
! OPTIONAL DEFAULTS (instead of READ IN CODE V PARAMETERS)
!
! ^zoom_pos == 1
! ^field_pos == 1
! ^wave_pos == 1
! ^global_sur == (sto z^zoom_pos)
!
!       READ IN CODE V PARAMETERS

ver all no
wri""
wri"Enter global surface number"
rea ^global_sur
wri""
wri"Enter zoom position"
rea ^zoom_pos
wri""
wri"Enter field position"
rea ^field_pos
wri""
wri"Enter Code V wavelength position"
rea ^wave_pos
wri""
wri Q"Code V dimension is 'c'" (dim)
wri""
out t compfile.out      !       write output to screen and default file compfile.out
```

```
!               chief ray direction @object surface

'1.chief == (l r1 so w^wave_pos g^global_surf^field_pos z^zoom_pos)
^m_chief . . (m r 1 so w^wave_pos g^global sur f ^field_ po s z^zoom_pos)
^n_chief == (n r] so w^wave_pos g^global_sur f^field_ pos z^zoom_ pos)

^x_s1 == (x r1 s1 w^wave_pos g'global. sur f'field_pos z^zoom_pos)
^y_s1 == (Y r1 s1 w^wave_ pos q^qlobal_sur f ^fiel d_ pos z^zoom_ pos)
^z_s1 ', (z r1 ?1 w^ wave_p os g^glo bal_ sur f^field_pos z' zoom_ pos)

!       FOR ' IN FIN ITE' CONJUGATE (de fined as lob ject dista nce| >= 1e7 units)

if (absf((thi so z^zoom_ pos)) >= (1.000e7) )

        ^z_source ==(thi so z^zoom_pos)
        ^aperture ==(epd)
!
!       - Locate CR 100 units before s1
!
        ^x_chief ==(^x_s1-(100*^l_chief))
        ^y_chief ==(^y_s1-(100*^m_chief))
        ^z_chief ==(^z_s1-(100*^n_chief))
els
!       FOR 'FINITE' CONJUGATE (defined as |object distance| < 1e8 units)

        ^aperture == absf(2*(nao))
!
        ^x_so ==(x r1 so w^wave_pos g^global_sur f^field_pos z^zoom_pos)
        ^y_so ==(y r1 so w^wave_pos g^global_sur f^field_pos z^zoom_pos)
        ^z_so ==(z r1 so w^wave_pos g^global_sur f^field_pos z^zoom_pos)
!
!       - ChfRayPos halfway between object and s1.

        if (thi so z^zoom_pos) > 0              ! REAL OBJECT
                ^z_source == -((thi so z^zoom_pos)/2)
                ^x_chief ==  ((^x_so+^x_s1)/2)
                ^y_chief ==  ((^y_so+^x_s1)/2)
                ^z_chief ==  ((^z_so+^x_s1)/2)
        els if (thi so z^zoom_pos) < 0          ! VIRTURAL OBJECT
                ^z_source == -(3*(thi so z^zoom_pos)/2)
                ^x_chief == -((^x_so+^x_s1)/2)
                ^y_chief == -((^y_so+^x_s1)/2)
                ^z_chief == -((^z_so+^x_s1)/2)
        els
                wri""
                wri"Cannot have zero object distances.  Increment thi so"
                wri"infinitesimally as an approximation so that rays travel
                wri"sequentially."
                wri""
        end if
end if

^format4 =="ChfRayDir= '2e.13e' '2e.13e' '2e.13e'"
^format5 =="ChfRayPos= '2e.13e' '2e.13e' '2e.13e'"

wri Q^format4 ^l_chief ^m_chief ^n_chief
wri Q^format5 ^x_chief ^y_chief ^z_chief

!               WRITE OTHER INITIAL PARAMETERS
!
^wavelength == (wl w wave_pos)
                                  convert CodeV wl in nm to system dimensions
if (dim) = 'I'                  ! inches
        ^wavelength == ^wavelength/2.54*1.000000000e-07
els if (dim) = 'C'              ! centimeters
        ^wavelength == ^wavelength*1.000000000e-07
els                             ! (DIM)='M' only other possibility, millimeters
        ^wavelength == ^wavelength*1.000000000e-06
end if

^format6  ==" zSource= '2e.13e'"
^format7  =="  IndRef= '2e.13e'"
^format8  =="  Extinc= '2e.13e'"
^format9  ==" Wavelen= '2e.13e'"
^format10 =="    Flux= '2e.13e'"
^format11 ==" GridType=  '2d'"
^format12 ==" Aperture= '2e.13e'"
^format13 ==" Obscratn= '2e.13e'"
^format14 ==" nGridpts=  '3d'"
^format15 =="    xGrid= '2e.13e' '2e.13e' '2e.13e'"
^format16 =="    yGrid= '2e.13e' '2e.13e' '2e.13e'"
^format17 =="     nElt=  '3d'"

wri Q^format6 ^z_source
wri Q^format7 ^n_source
wri Q^format8 ^ext_coef
wri Q^format9 ^wavelength
wri Q^format10 ^flux
wri Q^format11 ^gridtype
wri Q^format12 ^aperture
```

```
wri Q format13 ^obscuration
wri Q^format14 ^ngridpts
wri Q^format15 1 0 0
wri Q^format16 0 1 0
wri Q^format17(nums)
!
!^format1 =='3d'  '2e.13e'  '2e.13e'  '2e.13e' '2e.13e'  '2e.13e'  '2e.13e'"
^format18 .."       iElt=  '3d'"
^format19 . . .  EltName= surf. 'd'"
^format20 ==" EltType .   '3d'"
^format21 =="      fElt= '2e.13e'.
^format22 =="      eElt . '2e.13e'.
^format22a == " AsphCoef='2e .13e ' '2e. 13e' '2e.13e' '2e.13e' "
^format23 =="   psiElt='2e.13e' '2e.13e' '2e.13e' .
^format24 =="   VptElt= '2 e. 13e' '2e.13e' '2e. 13 e'"
^format25 =="   Rpt Elt= '2e.13e' '2e. 13e' '2e.13e'"
^format26 == "  IndRef =  '2e.13e'"
^format27 =="   Extinc= '2e.13e'
^format28 =="     zElt= '2e.13e'"
^format29 == " PropType=     '1d'"
^format30 =="  nECoord.    '2d'"

fol ^s 1 (nums)

^a_flat_surf == (absf((rdy s^s z^zoom_pos))>=^infinity)
^current_n  == ((ind s^s z^zoom_pos w^wave_pos))


if (rmd s^s z^zoom_pos)=, REFL,                    ! mirror
        if (typ sur s^s)='SPH'
                if (^a_flat_surf)
                        ^elt_type == 2             ! flat reflector
                els
                        ^elt_type == 1             ! conic relector
                end if
        els if (typ sur s^s)='CON'
                ^elt_type == 1                     ! conic relector
        els if ((typ sur s^s)='ASP')or((typ sur s^s)='YTO')  ! aspheric reflector or
                ^elt_type == 12                                ! y-toroid set to asphere
                wri""
                wri"ASPHERIC COEFFICIENTS FOR THE NEXT SURFACE ARE ONLY LISTED"
                wri"TO 10th ORDER 'D' TERM."
                wri""
                if (rdy s^s z^zoom_pos)>= 0
                        ^a_sur == (a s^s z^zoom_pos)
                        ^b_sur == (b s^s z^zoom_pos)
                        ^c_sur == (c s^s z^zoom_pos)
                        ^d_sur == (d s^s z^zoom_pos)
                els
                        ^a_sur == -(a s^s z^zoom_pos)
                        ^b_sur == -(b s^s z^zoom_pos)
                        ^c_sur == -(c s^s z^zoom_pos)
                        ^d_sur == -(d s^s z^zoom_pos)
                end if
        end if
!
els if (rmd s^s z^zoom_pos)='REFR'      ! refractive surface
        ^previous_n == (ind s^s-1 z^zoom_pos w^wave_pos)
        ^an_index_match == (^previous_n=^current_n)


        if (typ sur s^s)='SPH'
                if (^s=(num s))and(^a_flat_surf)
                        ^elt_type == 3    ! focal plane (if the image plane is curved
                els if (^an_index_match) ! it will not be defined as a focal plane.)
                        ^elt_type == 4  ! reference surface
                els
                        ^elt_type == 8  ! conic refractor
                end if
        els if (typ sur s^s)='CON'
                if (^an_index_match)
                        ^elt_type == 4  ! reference surface
                els
!                       ^elt_type == 8  ! conic refractor
                end if
        els if ((typ sur s^s)='ASP')or((typ sur s^s)='YTO')  ! aspheric refractor or
                ^elt_type == 20                              !y-toroid set to asphere
                wri""
                wri"ASPHERIC COEFFICIENTS FOR THE NEXT SURFACE ARE ONLY LISTED"
                wri"TO 10th ORDER 'D' TERM."
                wri""
                if (rdy s^s z^zoom_pos)>= 0
                        ^a_sur == (a s^s z^zoom_pos)
                        ^b_sur == (b s^s z^zoom_pos)
                        ^c_sur == (c s^s z^zoom_pos)
                        ^d_sur == (d s^s z^zoom_pos)
                els
                        ^a_sur == -(a s^s z^zoom_pos)
                        ^b_sur == -(b s^s z^zoom_pos)
                        ^c_sur == -(c s^s z^zoom_pos)
                        ^d_sur == -(d s^s z^zoom_pos)
```

```
                end if
        end if
else
        wri""
        wri"This macro does not incorporate this element type."
        wri"Surface data not included here."
        wri""
        got label_100
end if

!                       Calculate eccentricity and f
!
if (typ sur s^s) = 'SPH'
        ^e_elt= 0.000
        ^f_elt = absf((rdy s^s z^zoom_pos))
else if ((typ sur s^s)='ASP')or((typ sur s^s)='CON')or((typ sur s^s)='YTO')
        ^conic_const = (k s^s z^zoom_pos)
        if ^conic_const> 0                  ! oblate ellipsoid
                ^e_elt= sqrtf(^conic_const/(1.0+^conic_const))
                ^f_elt= absf((rdy s^s z^zoom_pos))/(1.000+^e_elt)
                wri""
                wri"Verify fElt, eElt for the following oblate ellipsoid surface."
                wri""
        else
                ^e_elt=sqrtf(-(^conic_const))
                ^f_elt=absf((rdy s^s z^zoom_pos))/(1.000+^e_elt)
        end if
end if
if (typ sur s^s)='YTO'
                wri""
                wri"Y-toroid has been modelled as an aspheric surface."
                wri"Verify/correct EltType and other data for the following surface."
                wri""
end if

        ^x_sur = (xsc s^s g^global_sur z^zoom_pos)
        ^y_sur = (ysc s^s g^global_sur z^zoom_pos)
        ^z_sur = (zsc s^s g^global_sur z^zoom_pos)
        ^l_sur = (lsc s^s g^global_sur z^zoom_pos)
        ^m_sur = (msc s^s g^global_sur z^zoom_pos)
        ^n_sur = (nsc s^s g^global_sur z^zoom_pos)

        wri Q^format18 ^s
        wri Q^format19 ^s
        wri Q^format20 ^elt_type
        wri Q^format21 ^f_elt
        wri Q^format22 ^e_elt
        if (^elt_type = 12)or(^elt_type = 20)
                wri Q^format22a ^a_sur ^b_sur ^c_sur ^d_sur
        end if
        if (rdy s^s z^zoom_pos) >= 0
                wri Q^format23 ^l_sur ^m_sur ^n_sur
        else
                wri Q^format23 -(^l_sur) -(^m_sur) -(^n_sur)
                                                reverse psi for negative element        end if
        wri Q^format24 ^x_sur ^y_sur ^z_sur
        wri Q^format25 ^x_sur ^y_sur ^z_sur
        wri Q^format26 absf(^current_n)
        wri Q^format27 ^ext_coef
        wri Q^format28 ^f_elt
        wri Q^format29 ^prop_type
        wri Q^format30 ^n_record

!       wri Q^format1 ^s ^x_sur ^y_sur ^z_sur ^l_sur ^m_sur ^n_sur       !table
label label_100
end for
!
!                       CALCULATE Tout

!       This calculates Tout for COMP.  Tout defines the image plane coordinate system
!       and path length, in global coordinates.  The path length, dL, is set to 1.
!
^num_sur = (num s)
!                               ! need to use (num s) since
!                               ! (typ dec s'i') doesn't work in conditional
if (typ dec s^num_sur)='DAR'    ! THIS ELIMINATES ERROR MESSAGES FOR NO DDA.
if ((ade si)<>0)or((bde si)<>0)or((cde si)<>0)
        ins si                  ! add surface to perform image plane tilts
        ade si-1 (ade si)       ! new surface is now si-1
        bde si-1 (bde si)       ! put image plane tilts on this surface
        cde si-1 (cde si)       ! since s decenters preceed tilts
end if
end if
!                       Evaluate image plane x-vector
!
xde si 0                        ! initialize decenters to zero
yde si 0
zde si 0

^initial_x = (xsc si g^global_sur z^zoom_pos)  ! determine origin
^initial_y = (ysc si g^global_sur z^zoom_pos)
```

```
^initial_z == (zsc si g^global_sur z^zoom_pos)

xde si 1                ! displace image plane to talc. x-d ir

^final_x == (xsc si g^global_ sur z^zoom_ pos)
^final_y == (ysc si g-global. sur z^zoom_pos)
^final_z == (zsc si g^global_ sur z^zoom_pos)

!       Calculate direction cosines for image plane x-axis
^lsc_x == ^final_x - ^initial_x          ! global l-dircosine
^msc_x == ^final_y - ^initial_y          ! global m-dircosine
^nsc_x == ^final_z - ^initial_z          ! global n-dircosine

xde si 0                ! re initialize x-decenter

!               Evaluate image plane y-vector

yde si 1                ! displace image plane to talc. y-dir

^final_x == (xsc si g^global_sur z^zoom_pos)
^final_y == (ysc si g^global_sur z^zoom_pos)
^final_z == (zsc si g^global_sur z^zoom_pos)
!
!       Calculate direction cosines for image plane y-axis
^lsc_y == ^final_x - ^initial_x          ! global l-dircosine
^msc_y == ^final_y - ^initial_y          ! global m-dircosine
^nsc_y == ^final_z - ^initial_z          ! global n-dircosine
!
!
^format2a== " nOutCord=     5"
^format2 == "      Tout= '2e.13e' '2e.13e' '2e.13e' '2e.13e' '2e.13e' '2e.13e' '2e.13e'"
^format3 == "            '2e.13e' '2e.13e' '2e.13e' '2e.13e' '2e.13e' '2e.13e' '2e.13e'"
!
wri Q^format2a
wri Q^format2 ^lsc_x ^msc_x ^nsc_x ^zero ^zero ^zero ^zero
wri Q^format3 ^lsc_y ^msc_y ^nsc_y ^zero ^zero ^zero ^zero
wri Q^format3 ^zero ^zero ^zero ^lsc_x ^msc_x ^nsc_x ^zero
wri Q^format3 ^zero ^zero ^zero ^lsc_y ^msc_y ^nsc_y ^zero
wri Q^format3 ^zero ^zero ^zero ^zero ^zero ^zero  1
wri""
wri""
wri""
out t               ! close output file, output to screen only

ver all yes
res                 ! restore original lens file
```

```
ver all no
!*  !dyt-zace.see]   Code V macro
!*
!*  Writes glob al coordinate ray trace
!*  to 10 sign ifican t figures.
!*
!*  Usage :                 in raytrace
!*  Pr ompts for :          Code V ray num ber
!*                          global surface numb er
!*                          zoom posi tier,
!*                          field position
!*                          wavelength posi t ion
!*
!*  Hir oshi Kadogawa
!*  Jet Propul sion Laborator y
!*  Pasaden a, CA
!*  e-mail:       hiro@osaserv.jpl.nasa.gov
!*
!*  7/29/93
!************** 4 *.*.*...*.. . ************* **** *********.....

! VARI ABLE DECI ARATIONS

chk yes                     ! enfor ce explici t decl arat i on
lcl str ^exp ^format1
l clnum ^ray_num ^glob al_ sur ^zoom_pos ^f ield_pos ^wave_p os ^s
lcl num ' index ^x_ pos ^y_p os ^z_pos ^l_ val ^m_val ^n_val !
! MAIN PROGRAM
!                           read in r equi r ed data
wri""
wr i "This progr am per fox-ins  a glob al coodin ate system raytrac e"
wri"to 10 sign ifican t figur es."
wri""
wr i "Enter Code V ray number"
r ea ^ray_num
Y?"m
wri"Ent er glob al sur face num b er .
rea ^global_sur
wri""
wr i"Ent er zoom posi tier, "
rea '-200[ [,.. pos
wri""
wri"Ent er field posi tion"
r ea ^field_pos
wri""
wr i "En ter Code V wavelengthposition"
rea ^wave_pos
wri""
                            show user other data
if (dim)='C'
        wri Q"Code V dimension is CENTIMETERS."
els if (dim)='M'
        wri Q"Code V dimension is MILLIMETERS."
els
        wr i Q" Code V dimen sion i s INCHES "
en d i f

wri""
wri Q" Lens title: '40c'" (tit)
wri""
wri Q"''11c'  '8 c'" (dat) (tim )
wri""
!                           as sign outp ut values
                            f orma toutput
wr i Q"sur face"
w r i Q "number        x                  Y                  7                  1                  m                  n"
wri"
^format1 = "'3d'  '2e.13e'  '2e.13e'  '2e.13e' '2 e.13e' '2e.13e' '2e.13e'"
for ^s 0 (num s)
        ^index = (ind s^s z^zoom_pos w^wave_pos)
        ^x_pos = (x r^ray_num s^s g^global_sur z^zoom_pos f^field_pos w^wave_pos)
        ^y_pos = (y r^ray_num s^s g^global_sur z^zoom_pos f^field_pos w^wave_pos)
        ^z_pos = (z r^ray_num s^s g^global_sur z^zoom_pos f^field_pos w^wave_pos)
        ^l_val = (l r^ray_num s^s g^global_sur z^zoom_pos f^field_pos w^wave_pos)/absf(^index)
        ^m_val = (m r^ray_num s^s g^global_sur z^zoom_pos f^field_pos w^wave_pos)/absf(^index)
        ^n_val = (n r^ray_num s^s g^global_sur z^zoom_pos f^field_pos w^wave_pos)/absf(^index)
   wri Q^format1 ^s  ^x_pos ^y_pos ^z_pos ^l_val ^m_val ^n_val
end for
!
ver all yes
```

Attachment 2

Code V Global Coordinate System Raytrace Macro

```
ver all no
!*****************************************************************
!* global_surf.seq -   code v macro
!*
!* Writes global surface coordinates with higher precision
!*than the .gsc command.
!*
!* Usage:                in global_surf
!* Prompts for :         global surface number
!*                       zoom position
!*
!* Hiroshi Kadogawa
!* Jet Propulsion Laboratory
!* Pasadena, CA
!* e-mail:       hiro@osaserv.jpl.nasa.gov
!*
!* 7/29/93
!*****************************************************************

! VARIABLE DECLARATIONS
!
chk yes                    ! enforce explicit declaration,
lcl str ^exp ^format1
lcl num ^s ^global_sur ^zoom_pos
lcl num ^x_sur ^y_sur ^z_sur ^l_sur ^m_sur ^n_sur
!
! MAIN PROGRAM

wri""
wri"This program provides global surface, coordinate system"
wri"positions and direction cosines"
wri"to 10 significant figures . ."
wri""
wri"Enter global surface number"
rea ^global_sur
wri""
wri"Enter zoom position"
rea ^zoom_pos
wri""
wri Q"Code V dimension is 'c'" (dim)
wri""
wri Q"surface"
wri Q"number       xsc                ysc                zsc                lsc                msc                nsc"
wri""
^format1 = "'3d'  '2e.13e'  '2e.13e'  '2e.13e' '2e.13e'  '2e.13e'  '2e.13e'"
for ^s 1 (num s)
        ^x_sur = (xsc s^s g^global_sur z^zoom_pos)
        ^y_sur = (ysc s^s g^global_sur z^zoom_pos)
        ^z_sur = (zsc s^s g^global_sur z^zoom_pos)
        ^l_sur = (lsc s^s g^global_sur z^zoom_pos)
        ^m_sur = (msc s^s g^global_sur z^zoom_pos)
        ^n_sur = (nsc s^s g^global_sur z^zoom_pos)
        wri Q^format1 ^s ^x_sur ^y_sur ^z_sur ^l_sur ^m_sur ^n_sur
end for
ver all yes
```

Attachment 3

Code V Global Surface Coordinate Macro