```
function xyz_to_magcoord, loc, input

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;; Computes the position of each packet in the torus coordinates M and zeta
;;
;; Inputs:
;;   * *loc.x, *loc.y, *loc.z = cartesian coordinates of packets (R_J)
;;   * phi = orbital longitude of packets (radians)
;;   * lam = magnetic longitude of packets (radians)
;;   * consts = list of magnetic dipole constants
;;   * plamsa_info = contains plasma torus information
;; Outputs:
;;   M = M shell (modified L shell) (R_J)
;;   zeta = distance along field line from centrifugal equator to packet (R_J)
;;   L = true L shell (R_J)
;;
;; Version History
;;   3.1: 4/27/2011
;;     * changing out_of_shadow -- does the planet but not moons
;;   3.0: 7/21/2010
;;     * Updating for new structure architecture
;;   -- 4/26/10 -- Added empty case statement 'Earth'
;;   2.0: 5/27/2009
;;     * Fixed issues with position in IPT
;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

common constants

case (input.geometry.planet) of
  'Mercury': magcoord = {out_of_shadow:ptr_new(0)}
  'Earth': magcoord = {out_of_shadow:ptr_new(0)}
  'Jupiter': begin
    magcoord = {L:ptr_new(0), M:ptr_new(0), zeta:ptr_new(0), lam:ptr_new(0), $
      out_of_shadow:ptr_new(0)}

    ;; See notes from 2008-05-13 for full description of this calculation.
    locx = (*loc.x)[*,0]
    locy = (*loc.x)[*,1]
    locz = (*loc.x)[*,2]
    phi = atan(-locx, locy)
    CML = input.geometry.cml - DipoleConsts.magrat*(*loc.t)  ;; current CML
    *magcoord.lam = CML - phi + !pi

    alpha = -DipoleConsts.tilt * cos(*magcoord.lam-DipoleConsts.lam3) ;angle of B equator

    ;;; Location of the dipole center in xyz
    lam_d = CML - DipoleConsts.offlong
    delx = DipoleConsts.offset * sin(lam_d)
    dely = -DipoleConsts.offset * cos(lam_d)
  delz = 0.

  ;; Positions relative to center of dipole
  xx = locx - delx
  yy = locy - dely
  zz = locz - delz

  ;; Account for E/W electric field
  r0 = sqrt(xx^2 + yy^2 + zz^2)
  xx -= input.plasma_info.eps*R0  ;; E/W electric field effectively moves packets east
  r1 = sqrt(xx^2 + yy^2 + zz^2) ;; Recompute distance from center

  ;;; Determine L
  orblat = asin(zz/r0)
  maglat = orblat - alpha
  centlat = orblat - 2./3.*alpha ;; centrifugal latitude
  *magcoord.L = r1 / (cos(maglat))^2
   ;; M = L * (cos(alpha/3.))^2 ;; M = dist from Jup that field line hits cent. eq.
   ;;; Don't actually want L since need the centrifugal equator
   ;; The Mag latitude of the centrifugal equator is alpha/3.
  *magcoord.M = *magcoord.L * cos(alpha/3.)^2

  ;; Determine zeta -- perp distance from packet to cent. equator
  *magcoord.zeta = r1 * sin(centlat)

  ;;;; Determine zeta -- old way
;; cos2lat = sqrt(5.-3*cos(2*latD))
;; cos2th = sqrt(5.-3*cos(2*theta))
;;
;; x1 = sqrt(6.)*sinlat/cos2lat
;; atanhx = .5 * alog((1.+x1)/(1.-x1))
;; analy1 = ( sqrt((coslat)^4 + 4.*(coslat)^2*(sinlat)^2 )) * $
;;   ( atanhx / (sqrt(6.) * coslat * cos2lat) + sinlat/coslat/2. )
;;
;; x2 = sqrt(6.)*sintheta/cos2th
;; atanhx = .5 * alog((1.+x2)/(1.-x2))
;; analy2 = ( sqrt((costheta)^4 + 4.*(costheta)^2*(sintheta)^2 )) * $
;;   ( atanhx / (sqrt(6.) * costheta * cos2th) + sintheta/costheta/2. )
;;
;; zeta = L * (analy1-analy2)
  end
 'Saturn': begin
  magcoord = {L:ptr_new(0), M:ptr_new(0), zeta:ptr_new(0), out_of_shadow:ptr_new(0)}
  r0 = sqrt( ((*loc.x)[*,0])^2 + ((*loc.x)[*,1])^2 + ((*loc.x)[*,2])^2)
  *magcoord.zeta = asin((*loc.x)[*,2]/r0) ;; magnetic latitude
  *magcoord.L = r0 / (cos(zeta))^2
  *magcoord.M = *magcoord.L
  end
 'Pluto': magcoord = {out_of_shadow:ptr_new(0)}
endcase
```

```
;; Check to see in packets are shadowed by planet or a moon
rho = sqrt((*loc.x)[*,0]^2 + (*loc.x)[*,2]^2)
*magcoord.out_of_shadow = ((rho GT 1) or ((*loc.x)[*,1] LT 0))

return, magcoord

end

```