```
pro quick_look, outfile, geomfile, image=image, x0=x0, imtype=imtype

if (n_elements(imtype) NE 1) then imtype = 'column'

;; If geofile isn't given then get it
if (n_elements(geomfile) NE 1) then stop

;; restore the outputs
restore, outfile
SystemConstants, run_info.planet, c

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; Print out some basic information
print, outfile
print, 'Planet: ' + run_info.planet
print, 'Starting Point: ' + run_info.startpoint
q = where(*run_info.gravity EQ 1, nq)
if (nq EQ 0) $
  then print, 'Gravity was not turned on' $
  else for i=0,nq-1 do print, (*c.objects)[q[i]] + '''s gravity is on'
print, 'Total run time = ' + strtrim(string(run_info.endtime/3600.), 2) + ' hours'
print, 'Neutral Species = ' + run_info.atom
print, 'Radiation Pressure is ' + ((run_info.radpres) ? 'on' : 'off')

if (run_info.fullsystem) $
  then print, 'Tracking full system' $
  else print, 'Only tracking packets within ' + $
    strtrim(string(run_info.outeredge),2) + ' object radii.'

print, '********************'
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; Show the initial velocity distribution
;;window, 0
wset, 0
show_veldist, proc_info, run_info, vrange=vrange, theo=theo, /disp

vv = sqrt(*startloc.vx^2 + *startloc.vy^2 + *startloc.vz^2)
mm = minmax(vrange) & dv = vrange[1]-vrange[0]
actual = histw(vv, *loc.frac, min=mm[0], max=mm[1], bin=dv)/dv

;plot, vrange, theo, xr=[0,15], /ylog, yr=[10,1e6], /xst
oplot, vrange, actual, color=2
xyouts, .55, .85, /norm, 'Initial Velocity Distribution!c  (all packets)'
xyouts, .55, .75, /norm, 'Initial Velocity Distribution!c  (remaining packets)', color=2

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;; Print out Loss Processes
if (run_info.lifetime EQ 0) then begin
  print, 'Loss Processes Included'
  for i=0,n_elements(*loss_info.reactions)-1 do $
    print, '  (' + strtrim(string(i),2) + ') ' + (*loss_info.reactions)[i]
```

```
    print, '**********************'
  endif else print, strtrim(string(round(run_info.lifetime/3600)), 2) + ' hour lifetime'

  ;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
  ;; Make a Column density image
  case (imtype) of
    'intensity': image = model_images('intensity', outfile, geomfile, 1., line='5890')
    'density': image = model_images('density', outfile, geomfile, 1., dz=0.1, zplane=0)
    else: image = model_images('column', outfile, geomfile, 1.)
  endcase

  restore, geomfile
  ;window, 1, xs=geoms.xs+150, ys=geoms.xs+150
  wset, 1

  xc = cos(findgen(361)*!dtor) & yc = sin(findgen(361)*!dtor)
  x0 =(findgen(geoms.xs)/(geoms.xs-1)-.5)*((geoms.xr)[1]-(geoms.xr)[0])/ $
    (*c.radius)[geoms.center]
  plot, findgen(10), /nodata, xr=minmax(x0), yr=minmax(x0), /xst, /yst, $
    xtit='Distance from ' + (*c.objects)[geoms.center] + ' (R!dObj!n)', $
    ytit='Distance from ' + (*c.objects)[geoms.center] + ' (R!dObj!n)', $
    pos=[100,100,100+geoms.xs,100+geoms.ys], /dev, tit=file_basename(outfile)
  disparr, image, 3, /log, result=image2, /nodisp, low=l, high=h
  tv, bytscl(image2, l, h, top=220)+35, 100, 100, /dev
  polyfill, xc, yc, color=4
  plots, [100,100,100+geoms.xs,100+geoms.xs,100], [100,100+geoms.ys,100+geoms.ys,100,100],$
    /dev

  @destroy_all
  destroy_constants, c

  end
```