```
function [stateseen, act_taken, rew] =
gen_rt_episode(ei,pol_pi,RT,posStarts,nPosStarts,maxNStates,maxNActions,maxNPii,maxNPjj,m
axNVii,maxNVjj,maxNAii,maxNAjj)
% GEN_RT_EPISODE - Generates a RT episode
%
% Note: this version is not ness. very strict with regards to whether we jump over
% corners of our track.  A better version would calculate the furthest we could go
% before we intersect the edge and then use that as a starting point for the next
iteration.
%
% In any event this version does provide an "environment" even if somewhat strange in
which
% our reinforcement algorithm can operate and the needed modifications to make this more
% realistic done at any time.
%
% Written by:
% --
% John L. Weatherwax                2007-12-07
%
% email: wax@alum.mit.edu
%
% Please send comments and especially bug reports to the
% above email address.
%
%-----

PLT_SPS=0; PRNT_STUFF=0;

if( PLT_SPS )
  figure; imagesc(RT); colorbar; hold on;
end

rew=0; stateseen = []; act_taken = [];

% pick an initial starting state, position and velocity (using exploring starts):
%ii=maxNPii; tmp = randperm(nStarts); jj=posStarts(tmp(1)); clear tmp; vii = 0; vjj = 0;

% the initial position:
pii=maxNPii; pjj=posStarts(mod(ei,nPosStarts)+1);
% the initial velocity:
vii=mod(ei,maxNVii); vjj=mod(ei,maxNVjj);
% correct (vii,vjj) if we happen to have selected (0,0):
if( vii==0 && vjj==0 )
  if( unidrnd(2)==1 )
    vii=mod(ei,maxNVii-1)+1;
  else
    vjj=mod(ei,maxNVjj-1)+1;
  end
end

if( PLT_SPS ) plot(pjj,pii,'x'); end;

% accumulate/store the first state seen:
stateseen(1,:) = [ pii,pjj,vii,vjj ];

% implement a full episode following the policy specified by pol_pi:
while( 1 ) %~didWeFinish([pii,pjj,vii,vjj],maxNPjj) ) % take a step
  stInd       = sub2ind( [maxNPii,maxNPjj,maxNVii,maxNVjj], pii,pjj,vii+1,vjj+1 );
  reshape(pol_pi(stInd,:),[3,3]);
```

```
act_to_take = sample_discrete( pol_pi(stInd,:), 1, 1 );
act_taken   = [ act_taken; act_to_take ];
[aIndii,aIndjj] = ind2sub( [ maxNAii,maxNAjj ], act_to_take );
aii = aIndii-2; ajj = aIndjj-2; % the specific actions to take \in {-1,0,+1}
% update our state according to this action and recieve a reward:
vii=vii+aii;
vjj=vjj+ajj;
if( vii<0 || vii>5 )
  [pii,pjj,vii-aii,vjj-ajj,aii,ajj]
  error( 'vii out of bounds' );
end
if( vjj<0 || vjj>5 )
  [pii,pjj,vii-aii,vjj-ajj,aii,ajj]
  error( 'vjj out of bounds' );
end
pii=pii-vii;
pjj=pjj+vjj;

if( didWeFinish([pii,pjj,vii,vjj],maxNPjj) ) break; end

% add a random VALID component to our step:
rndUp=0; rndRt=0;
if( rand < 0.5 ) % we have a random step
  if( rand < 0.5 ) % that is up
    pii=pii-1; if( pii>0         ) rndUp=1; else, pii=pii+1; end
  else             % that is right
    pjj=pjj+1; if( pjj<maxNPjj+1 ) rndRt=1; else, pjj=pjj-1; end
  end
end
%--
% Now the "environment" responds to this action taken from this state:
%--
if( PRNT_STUFF )
  fprintf('[pii,pjj]=[%d,%d]...\n',pii,pjj);
  fprintf('onRT(pii,pjj)=%d...\n',onRT(pii,pjj,RT,maxNPii,maxNPjj) );
  fprintf('onRT(pii-1,pjj)=%d...\n',onRT(pii-1,pjj,RT,maxNPii,maxNPjj) );
  fprintf('onRT(pii,pjj+1)=%d...\n',onRT(pii,pjj+1,RT,maxNPii,maxNPjj) );
  fprintf('onRT(pii-1,pjj+1)=%d...\n',onRT(pii-1,pjj+1,RT,maxNPii,maxNPjj) );
end
if( onRT(pii,pjj,RT,maxNPii,maxNPjj) )
  rew = rew-1;
else
  rew = rew-5;
  % adjust our position if we fall off the track:
  % first obtain our original position (which was valid):
  pii = pii + vii; if( rndUp ) pii=pii+1; end
  pjj = pjj - vjj; if( rndRt ) pjj=pjj-1; end
  if( PRNT_STUFF )
    fprintf('[pii,pjj]=[%d,%d]...\n',pii,pjj);
    fprintf('onRT(pii,pjj)=%d...\n',onRT(pii,pjj,RT,maxNPii,maxNPjj) );
  end
  % adjust our velocity back to what we originally had:
  %vii=vii-aii; vjj=vjj-ajj;

  % find a valid next spot:
  if( onRT(pii-1,pjj,RT,maxNPii,maxNPjj) )
    pii=pii-1;
  elseif( onRT(pii,pjj+1,RT,maxNPii,maxNPjj) )
    pjj=pjj+1;
  elseif( onRT(pii-1,pjj+1,RT,maxNPii,maxNPjj) )
```

```
      pii=pii-1; pjj=pjj+1;
    else
      %error( 'can''t find a valid next spot' );
      % as a last ditch we will continue from our original spot (which by assumption was
valid)
      % sometimes we can fall of fthe RT by way of finishing ...
      % do nothing ... (pii, pjj) are set to a valid spot
    end
  end

  if( PLT_SPS ) plot(pjj,pii,'x'); end;

  stateseen(end+1,:) = [ pii,pjj,vii,vjj ];
end

return;



function onQ = onRT(pii,pjj,RT,maxNPii,maxNPjj)
% onRT -
%
if( ~( (1<=pii) && (pii<=maxNPii) ) )
  onQ = 0;
  return;
end
if( ~( (1<=pjj) && (pjj<=maxNPjj) ) )
  onQ = 0;
  return;
end
if( RT(pii,pjj)==1 )
  onQ = 1;
  return;
else
  onQ = 0;
  return;
end



function finishQ = didWeFinish(st,maxNPjj)
% DIDWEFINISH -
%
pii = st(1); pjj = st(2);

if( pjj>=maxNPjj )
  finishQ = 1;
else
  finishQ = 0;
end
```