Field D* Pseudocode

Algorithm 1 ComputeCost(s, s_a, s_b)

---

if (s_a is a diagonal neighbor of s) **then**
    $s_1 = s_b$; $s_2 = s_a$;
else
    $s_1 = s_a$; $s_2 = s_b$;
end if
c is traversal cost of cell with corners s, s_1, s_2;
b is traversal cost of cell with corners s, s_1, but not s_2;
if ($\min(c, b) = \infty$) **then**
    $v_s = \infty$;
else if ($g(s_1) \leq g(s_2)$) **then**
    $v_s = \min(c, b) + g(s_1)$;
else
    $f = g(s_1) - g(s_2)$;
    if ($f \leq b$) **then**
        if $c \leq f$ **then**
            $v_s = c\sqrt{2} + g(s_2)$;
        else
            $y = \min(\frac{f}{\sqrt{c^2 - f^2}}, 1)$;
            $v_s = c\sqrt{1 + y^2} + f(1 - y) + g(s_2)$;
        end if
    else
        if ($c \leq b$) **then**
            $v_s = c\sqrt{2} + g(s_2)$;
        else
            $x = 1 - \min(\frac{b}{\sqrt{c^2 - b^2}}, 1)$;
            $v_s = c\sqrt{1 + (1 - x)^2} + bx + g(s_2)$;
        end if
    end if
end if
return v_s;

Algorithm 2 ComputeShortestPath()

while $\min_{s \in OPEN}(key(s) < key(s_{start})$ OR $rhs(s_{start}) \neq g(s_{start}))$ **do**
  peek at node s with the minimum key on $OPEN$;
  if $(g(s) \geq rhs(s))$ **then**
    $g(s) = rhs(s)$;
    remove s from $OPEN$;
    for all $s' \in nbrs(s)$ **do**
      if (s' was not visited before) **then**
        $g(s') = rhs(s') = \infty$;
      end if
      $rhs_{old} = rhs(s')$;
      if $(rhs(s') >$ ComputeCost$(s', s, ccknbr(s', s)))$ **then**
        $rhs(s') =$ ComputeCost$(s', s, ccknbr(s', s))$;
        $bptr(s') = s$;
      end if
      if $(rhs(s') >$ ComputeCost$(s', s, cknbr(s', s)))$ **then**
        $rhs(s') =$ ComputeCost$(s', cknbr(s', s), s)$;
        $bptr(s') = cknbr(s', s)$;
      end if
      if $(rhs(s) \neq rhs_{old})$ **then**
        $UpdateState(s')$;
      end if
    end for
  else
    $rhs(s) = \min_{s' \in nbrs(s)}$ComputeCost$(s, s', ccknbr(s, s'))$;
    $bptr(s) = \text{argmin}_{s' \in nbrs(s)}$ComputeCost$(s, s', ccknbr(s, s'))$;
    if $(g(s) < rhs(s))$ **then**
      $g(s) = \infty$;
      for all $s' \in nbrs(s)$ **do**
        if $(bptr(s') = s$ OR $bptr(s') = cknbr(s', s))$ **then**
          if $(rhs(s') \neq$ComputeCost$(s', bptr(s'), ccknbr(s', bptr(s'))))$ **then**
            if $(g(s') < rhs(s')$ OR $s' \notin$ OPEN) **then**
              $rhs(s') = \infty$;
              UpdateNode(s');
            else
              $rhs(s') = \min_{s'' \in nbrs(s')}$ComputeCost$(s', s'', ccknbr(s', s''))$;
              $bptr(s') = \text{argmin}_{s'' \in nbrs(s')}$ComputeCost$(s', s'', ccknbr(s', s''))$;
              UpdateNode(s');
            end if
          end if
        end if
      end for
    end if
    UpdateNode(s);
  end if
end while