```
def linePosition3d(point=None, line=None):
    import numpy as np
    import math as m
    #LINEPOSITION3D Return the position of a 3D point projected on a 3D line
    #
    #   T = linePosition3d(POINT, LINE)
    #   Computes position of point POINT on the line LINE, relative to origin
    #   point and direction vector of the line.
    #   LINE has the form [x0 y0 z0 dx dy dy],
    #   POINT has the form [x y z], and is assumed to belong to line.
    #   The result T is the value such that POINT = LINE(1:3) + T * LINE(4:6).
    #   If POINT does not belong to LINE, the position of its orthogonal
    #   projection is computed instead.
    #
    #
    #   See also:
    #   lines3d, createLine3d, distancePointLine3d, projPointOnLine3d
    #
    #   ---------
    #   author : David Legland
    #   INRA - TPV URPOI - BIA IMASTE
    #   created the 17/02/2005.
    #

    #   HISTORY
    #   05/01/2007 update doc
    #   28/10/2010 change to bsxfun calculation for arbitrary input sizes
    #       (Thanks to Sven Holcombe)
    #   06/07/2018 translated to python by Valerie Martin


    # vector from line origin to point
    dp = np.subtract(point, line[0:3])

    # direction vector of the line
    vl = line[3:6]

    # precompute and check validity of denominator
    denom = np.dot(vl, np.transpose(vl))
    invalidLine = denom < 2**(-52)
    if invalidLine:
        denom = 1


    # compute position using dot product normalized with norm of line vector.
    pos = ((np.dot(dp, vl)/ denom))


    return pos

    # position on a degenerated line is set to 0
    pos[invalidLine] = 0
```