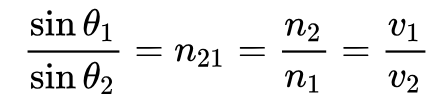
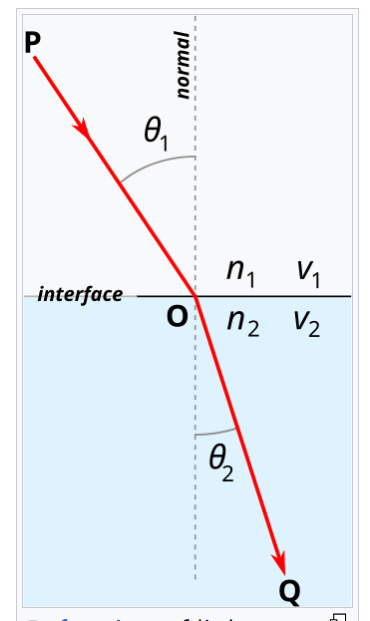
Our problem is to show how light wave is acting when it hits from one medium to another medium, with polarization analysis included.

This procedure contains 2 part:

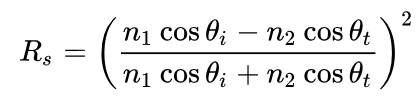
1. Snell’s law shows the direction of the light after reflection and refraction.
2. Fresnel Equation shows the energy relationship of s- and p- polarized electromagnetic wave when it hits at the interface. (reflectivity, transmissivity, and so on)

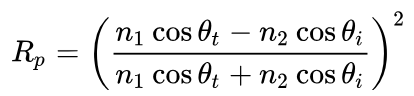
First we can achieve the Snell’s law, which gives the direction of the reflected and transmitted light.



Then, in Fresnel Equations, reflectivity is the ratio of power in and out, and this power need to be visualized using the brightness of incident ray in Unity. **This visualization procedure may be self-defined.**

**How to map brightness of ray and power of light could be an important question.**







14.02.2025：

Fresnel diffraction:

This part includes wave optics.

Pseudocode of Fresnel diffraction integral:

for 观察点x in 所有x坐标:

for 观察点y in 所有y坐标:

总振幅 = 0

for 孔径点x' in 孔径内:

for 孔径点y' in 孔径内:

dx = x - x'

dy = y - y'

相位 = (dx² + dy²) \* k / (2\*z)

总振幅 += 初始振幅[x',y'] \* exp(i\*相位)

光强[x,y] = |总振幅|²

# Fresnel Diffraction Integral (Pseudocode)

# ----------------------------------------

# λ: wavelength of light

# z: propagation distance

# k = 2π/λ: wavenumber

# aperture: 2D array representing initial field (1 = open, 0 = blocked)

for obs\_x in all\_x\_coordinates: # Loop through observation plane x-axis

for obs\_y in all\_y\_coordinates: # Loop through observation plane y-axis

total\_amplitude = 0 + 0j # Complex amplitude accumulator

# Integrate over aperture

for ap\_x in aperture\_x\_range: # Loop through aperture x-coordinates

for ap\_y in aperture\_y\_range:

if aperture[ap\_x, ap\_y] == 1: # Only calculate open areas

dx = obs\_x - ap\_x

dy = obs\_y - ap\_y

# Fresnel phase term calculation

phase = (dx\*dx + dy\*dy) \* k / (2\*z)

# Accumulate complex amplitude

total\_amplitude += initial\_amplitude[ap\_x, ap\_y] \* cmath.exp(1j \* phase)

# Calculate final intensity

intensity[obs\_x, obs\_y] = abs(total\_amplitude) \*\* 2

These steps has been went through in the mind and the important thing is to calculate the phase on a given propagation distance z. When we change z, in 2D we can get this figure given by Johnannes:

