UNIVERSITA' POLITECNICA DELLE MARCHE

FACOLTA' DI INGEGNERIA

Corso di Laurea triennale in Ingegneria Meccanica

Comfort Air: calibrazione del sensore di luminosità

Comfort Air: calibration of the brightness sensor

Relatore:

Prof. Gian Marco Revel

Correlatore:

D.ssa Serena Serroni

Tesi di Laurea di:

Federico Tottone

A.A. 2019 / 2020

Indice

Introduzione

È indubbio constatare che gli ambienti indoor, ovvero confinati, siano i luoghi predominanti per il trascorso delle nostre giornate: è stimato che ogni individuo trascorra circa il 90% della giornata in ambienti interni come abitazioni, luoghi di lavoro, di ristoro e di pratica fisica.

Ricreare spazi biocompatibili con un microclima idoneo e che non presenti anomalie che possano influire negativamente sulla salute, sulla sfera emotiva e sul benessere degli individui diviene pertanto fondamentale e vitale al fine di garantire una buona salute psicofisica ed anche per promuovere un aumento della produttività.[1]

Ad aspetti sociali, di comunicazione o meramente costruttivi si affiancano aspetti legati alla certificazione e alle norme vigenti per l'abitare sostenibile.

In questo caso, il certificato energetico, richiesto dalla normativa nazionale, rende trasparenti i futuri costi energetici, facilita le decisioni di acquisto o di affitto e offre ai proprietari degli immobili la possibilità di programmare per tempo gli investimenti per il risparmio energetico. [1]

Dunque, per comfort ambientale o **benessere ambientale indoor** (*IEQ – Indoor Environmental Quality*) si intende il raggiungimento di diversi tipi di benessere all'interno di un ambiente chiuso, quali:

- **benessere termo-igrometrico** (temperatura, umidità e velocità dell'aria);
- **benessere respiratorio-olfattivo** (qualità dell'aria);
- **benessere acustico** (livello di rumorosità);
- **benessere visivo** (livello di luminosità).

Un passo fondamentale per implementare l'efficienza energetica degli edifici e quindi, in generale, il benessere ambientale indoor, è quello di portare culture e tecnologie sostenibili nelle costruzioni.

A tal proposito, l'innovazione tecnologica, proposta in questa trattazione, è il ***Comfort Eye***, un sensore che permette di acquisire ed analizzare dati inerenti a diverse variabili (quali ad esempio l'illuminazione, la temperatura dell'aria, la quantità di CO_2), così da poterli utilizzare successivamente per apportare modifiche e rinnovamenti efficaci ed intelligenti all'interno dell'edificio, al fine di migliorare il benessere negli ambienti indoor, pur rispettando sempre le norme *UNI 10380/A1*, *UNI 10840*.

Entrando più nel particolare, nel corso di questa esposizione, è stato preso in considerazione il **benessere visivo** ed in dettaglio l'obiettivo del mio lavoro è stato quello di calibrare un sensore di conversione dell'intensità luminosa, installato all'interno di un dispositivo per la misurazione delle

variabili indoor, chiamato ***Comfort Air***, al fine di analizzare l'incertezza della misurazione stessa, per il monitoraggio del comfort visivo indoor.

Sintetizzando, in questa disamina, nel primo capitolo verranno descritti i due nodi che costituiscono il sensore ***Comfort Eye***, l'architettura generale di funzionamento di quest'ultimo ed infine una descrizione generale sulle caratteristiche e la normativa inerente al comfort visivo.
Successivamente, nel secondo capitolo, verranno richiamate alcune grandezze illuminotecniche, indispensabili per le analisi del fattore luce e per i susseguenti approfondimenti inclusi nel terzo capitolo, che riguarderanno il **sensore *TSL2591*** ed i suoi relativi fotodiodi.
Infine, nel quarto ed ultimo capitolo, verranno descritte ed illustrate le attività svolte in laboratorio, a partire dalla realizzazione del "set-up", alla successiva fase di acquisizione dei dati ed allo studio degli stessi per giungere alla fase di calibrazione, con relativo calcolo del grado di incertezza del sensore di luminosità, appartenente al dispositivo per la misurazione delle variabili indoor (***Comfort Air***), uno dei due nodi del sensore ***Comfort Eye***.

Capitolo 1

Sistema Comfort Eye

Il termine "***deep renovation***" è definito, secondo la direttiva EU sull'efficienza energetica, come una soluzione, economicamente vantaggiosa, di ristrutturazione di un edificio, al fine di ridimensionare notevolmente il consumo finale di energia rispetto alla condizione precedente all'intervento. [2]

Il ***progetto europeo P2Endure*** ("*Plug-and-Play product and process innovation for Energy-efficient building deep renovation*") promuove soluzioni innovative per il "*deep renovation*", basate su sistemi prefabbricati di tipo *Plug-and-Play*, abbinati con tecnologie innovative per la verifica e la realizzazione dell'intervento, quali stampa 3D, tecniche di diagnostica avanzata e "*Building Information Modeling*" (BIM).

Una possibile soluzione è data dal ***Comfort Eye*****,** un apparato a basso costo per il monitoraggio real-time delle condizioni di comfort negli ambienti; in sostanza, il sistema è basato su un **sensore IR (infrarossi)**, installato sul soffitto dell'ambiente ed azionato per conseguire la scansione dell'ambiente stesso.

Il sistema ***Comfort Eye*** inoltre è in grado di fornire il monitoraggio continuo del ***PMV*** (***Predicted Mean Vote***), ovvero un indice che esprime un voto medio assegnato ad un ambiente confinato, e la temperatura media radiante per più posizioni nell'ambiente: a quel punto vi sarà data la possibilità di introdurre i dati relativi al profilo di utilizzo dell'area in esame. [3]

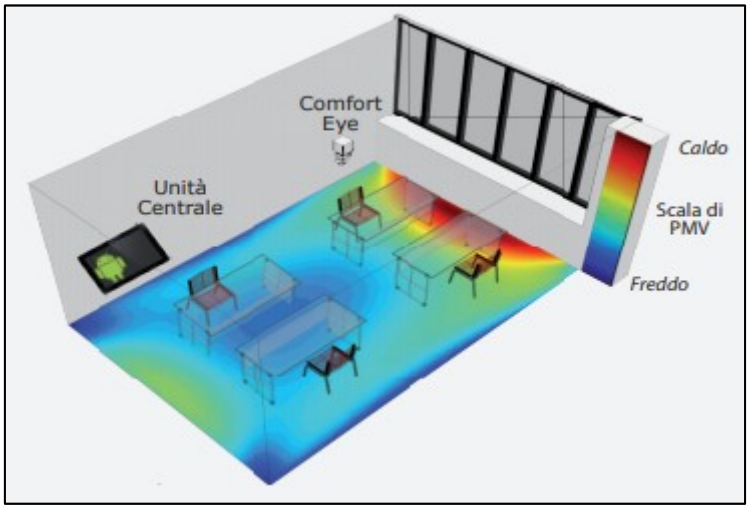

Figura 1 - Schema generico del sistema di monitoraggio del PMV

Un **microcontrollore embedded**, ovvero non riprogrammabile dall'utente per altri scopi, permette di amministrare i dati acquisiti da tale sensore, insieme a quelli degli altri sensori ambientali installati con l'unità di controllo, consentendo di elaborare i dati stessi e calcolare gli indici di comfort termico in più punti della stanza.

In particolare, sono due le componenti che caratterizzano l'utenza: l'attività svolta ed il tipo di abbigliamento. Questi due fattori vengono inseriti in fase di configurazione del sistema durante la prima installazione, ma possono essere aggiornati in qualsiasi momento anche dagli stessi occupanti, rendendoli parte attiva del sistema di misura. [2]

Il monitoraggio dato dal *Comfort Eye* potrà assicurare la misurazione di diverse variabili, con le relative incertezze, richieste ed auspicabili, riportate nella *tabella 1.*

Tabella 1 - Variabili ed incertezze del Comfort Eye

Variabile	Precisione	
	Richiesta	**Desiderata**
Temperatura dell'aria	$\pm$ 0.5 °C	$\pm$ 0.2 °C
Temperatura media radiante	$\pm$ 2 °C	$\pm$ 0.2 °C
Umidità assoluta espressa come pressione parziale di vapore acqueo	$\pm$ 0.15 kPa	$\pm$ 0.15 kPa
Velocità dell'aria	$\pm\frac{(0.05+0.05\text{va})m}{s}$	$\pm\frac{(0.02+0.07\text{va})m}{s}$
CO_2	$\pm$ 50 ppm	$\pm$ 30 ppm
Rumorosità	$\pm$ 1.1 dB	$\pm$ 0.7 dB
Illuminazione	<6%	<4%

1.1 Dispositivo di scannerizzazione ad infrarossi

Il *Comfort Eye* è costituito da tre nodi, due dei quali impiegati per il rilevamento e l'acquisizione dei dati richiesti ed un *gateway*, connesso ad un server in remoto, per il salvataggio dei dati stessi.

Il primo nodo è un **dispositivo di scannerizzazione ad infrarossi**, il quale determina una mappatura termica dell'ambiente indoor che è stato preso in esame.

Figura 2 - Dispositivo di scannerizzazione ad infrarossi

Questo dispositivo viene collocato sul soffitto della stanza valutata ed ha due gradi di libertà, dovuti alle rotazioni attorno a due assi.
Nel momento in cui viene scannerizzato l'ambiente, il dispositivo di scannerizzazione ad infrarossi spedisce le informazioni al *gateway*, un dispositivo di rete che funziona come punto di ingresso tra una rete e un'altra; il *gateway* consentirà di calcolare la **temperatura media radiante**, ovvero la temperatura di un ambiente fittizio, caratterizzato da pareti aventi tutte la stessa temperatura, nel quale un soggetto scambierebbe per irraggiamento lo stesso flusso termico che scambia nell'ambiente reale in cui ogni parete ha una propria temperatura (tale grandezza si misura in °C). [4]

Questa soluzione comporta due vantaggi molto interessanti da un punto di vista pratico:

1. Viene utilizzato un solo sensore per l'acquisizione di maggiori punti di misurazione della temperatura media radiante nell'ambiente e in quest'ultimo verrà selezionato il punto migliore per il calcolo della temperatura media radiante.

2. Le mappature termiche dell'ambiente indoor ottenute sono impiegabili per tracciare il rendimento dell'isolamento dell'edificio, la cui analisi è essenziale per assicurare elevate condizioni di comfort negli spazi abitati, sia negli interventi di nuova costruzione che in quelli di riqualificazione dell'esistente.

1.2 Dispositivo per la misurazione delle variabili indoor

Il secondo nodo, appartenente al *Comfort Eye*, è un **dispositivo per la misurazione delle variabili indoor**, chiamato ***Comfort Air***.

Figura 3 - Sensore Comfort Air

Esso è un dispositivo da scrivania, alimentato a 5V, che trova installati diversi sensori al suo interno, inerenti all'acquisizione dei seguenti parametri:

- **Temperatura dell'aria** [$°C$]
- **Umidità relativa** [kPa]
- **CO_2** [ppm]
- **TVOC - Total Volatile Organic Compounds** [ppm]
- **Illuminazione** [lux]

Quando la batteria, collocata all'interno del *Comfort Air*, è utilizzata per l'alimentazione, il sensore per l'acquisizione del parametro CO_2 viene interrotto poiché si procreerebbe un eccessivo dispendio energetico.

Pertanto, il *Comfort Air*, tramite l'ausilio dei sensori al suo interno, acquisisce i dati a disposizione durante l'analisi e, tramite l'equipaggiamento con un modulo *Wi-Fi*, spedisce i dati al *gateway*.

1.3 Gateway

Lo scopo del **gateway** è quello di veicolare i pacchetti di rete all'esterno di una rete locale (LAN) e, proprio per tale questione, richiede una connessione Internet 4G o una connessione attraverso il network locale.

Ai fini dell'analisi di questo lavoro, il gateway è stato sviluppato specificatamente per essere coerente al progetto *P2Endure*, affinché possa sostenere un monitoraggio a lungo termine.

Per tutto ciò, la sua funzione di lavoro viene svolta da un commerciale ***Raspberry Pi 0***, eccellente dal punto di vista del prezzo di acquisto sul mercato, mantenuto basso grazie all'utilizzo di una RAM integrata da 512*Mb* e di un processore *ARM single-core* da 1*GHz*. Inoltre, dispone di uno slot per micro-SD card push-pull, due porte micro-USB (una per l'alimentazione, una per la connessione USB) e una porta mini-HDMI. [5]

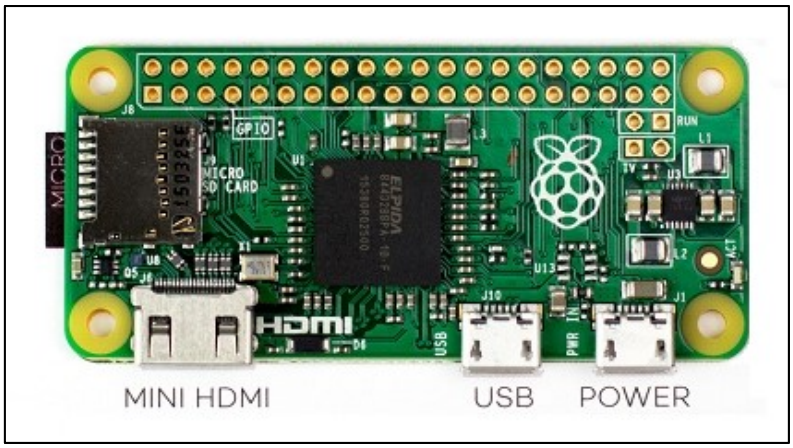

Figura 4 - Raspberry Pi 0

1.4 Architettura generale di funzionamento

La *figura 5* mostra l'architettura generale di funzionamento: anzitutto, i due nodi principali, ovvero il **dispositivo di scannerizzazione ad infrarossi** e il **dispositivo per la misurazione delle variabili indoor** hanno il compito di rilevare ed acquisire i dati (divisi in dati grezzi, cioè non lavorati, e in log operativi, ovvero registrazioni sequenziali e cronologiche delle operazioni effettuate dal sistema informatico) ed il **gateway**, connesso con i nodi appena citati, li memorizza.

A questo punto, il gateway, allacciato ad una rete internet, comunica con un server in remoto, individuato nei laboratori dell'Università Politecnica delle Marche (*UNIVPM*), dove, ogni 15 minuti, i dati vengono spediti elaborati e memorizzati in un database chiamato "***MySql***".

In questo modo si fornirà una pulizia dei dati grezzi ed una migliore mappatura termica; in rapida successione, i dati elaborati verranno messi a disposizione dall'utente, il quale potrà visualizzarli e ispezionarli tramite un'applicazione, usufruibile da ogni browser.

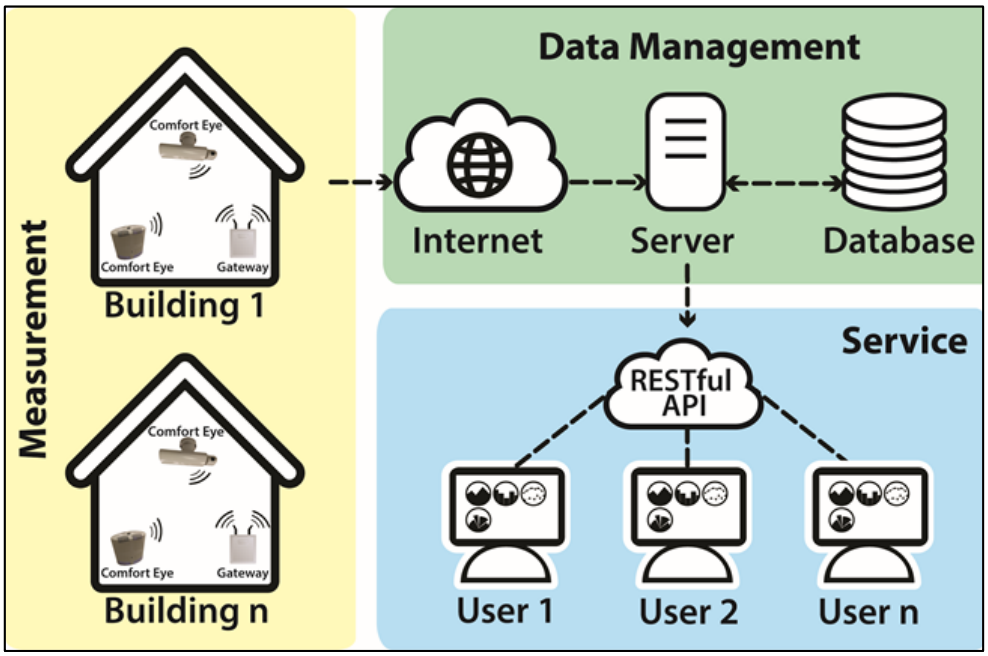

Figura 5 - Schema generale di funzionamento del progetto

1.5 Comfort visivo

L'illuminazione naturale può rappresentare una risorsa fondamentale durante la fase di progettazione e di realizzazione degli edifici, affinché essi possano risultare energeticamente sostenibili e qualitativamente confortevoli. Inoltre, un'efficace illuminazione diurna ed un'integrazione coerente di luce naturale e/o proveniente da sorgenti artificiali possono garantire in maniera significativa ad un sostanziale risparmio energetico negli edifici.

Allo stesso tempo, però, un'eccessiva disponibilità di luce naturale, particolarmente elevata, può creare alcuni svantaggi significativi negli ambienti indoor; ad esempio, per le condizioni climatiche, un elevato irraggiamento può implicare un incremento rilevante dei carichi frigoriferi e dunque un

conseguente incremento del fabbisogno energetico durante la fase di climatizzazione. Queste considerazioni sono necessarie, ad esempio, per prevedere e studiare adeguati sistemi di ombreggiamento per il periodo estivo.

L'illuminazione all'interno di un ambiente, sia essa artificiale o naturale, è considerata la principale responsabile del comfort visivo; in particolare l'illuminazione di un ambiente deve soddisfare alcune funzioni fondamentali quali le condizioni di sicurezza e le prestazioni visive (infatti, un'inadeguata illuminazione causa sensazione di malessere generale e stanchezza agli occhi). [6]

A tal proposito, nel 2011 il Consiglio Europeo in accordo con il Parlamento Europeo ha emanato l'ultimo aggiornamento alla norma del 2008 titolo "*Light and Lighting – Lighting of work places – Part 1: Indoor work places*" come riferimento normativo per tutti i paesi dell'Unione Europea in merito al livello di illuminazione suggerito negli ambienti di lavoro. Tale norma è stata poi recepita dall'Italia (sempre nel 2011) con la normativa ***UNI EN 12464-1*** allo scopo di prescrive quelli che debbono essere i requisiti illuminotecnici per gli interni di lavoro, al fine di garantire le prestazioni ed il comfort visivo in essi richiesti. [7]

Di seguito viene proposta una tabella con i valori di illuminamento consigliati:

Tabella 2 - Valori lux consigliati in ambienti indoor

LUOGHI	LUX CONSIGLIATI
Salotto	150-200 lux
Soggiorno	150-200 lux
Cucina	200-250 lux
Camera da letto	100-150 lux
Bagno	100-150 lux
Corridoi, scale	50-100 lux
Uffici	250-500 lux
Scuole	250-500 lux
Negozi	250-500 lux
Grandi magazzini	500-1000 lux

Capitolo 2

Grandezze illuminotecniche

Nella progettazione di un ambiente indoor, l'**illuminotecnica** assume un ruolo di rilievo in quanto si occupa dell'illuminazione degli spazi, sia interni che esterni, usufruendo sia della luce solare sia di quella artificiale. Gli studi realizzati fino ad oggi hanno dimostrato che questa disciplina è divenuta uno strumento eccellente per la bio-architettura, poiché consente di ottenere un utilizzo ottimale della luce solare e quindi contribuisce notevolmente al risparmio energetico limitando i consumi di energia elettrica. [8]

I fondamenti di illuminotecnica definiscono che le radiazioni visibili dall'occhio umano corrispondono ad una porzione limitata dello spettro elettromagnetico (non è un caso che le lunghezze d'onda che possiamo vedere ad occhio nudo siano quelle a cui il Sole irraggia con maggiore intensità), in un intervallo di lunghezze d'onda comprese tra circa 380 *nm* e circa 780 *nm* (*figura 6*).

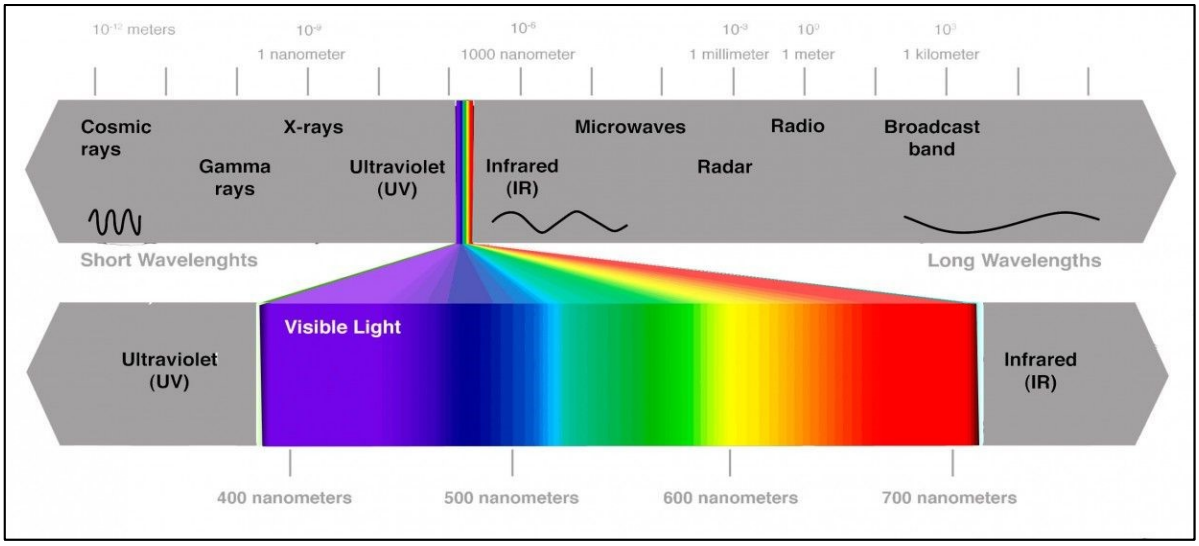

Figura 6 – Intervallo dello spettro elettromagnetico visibile dall'occhio umano

L'intervallo di lunghezze d'onda appena descritto corrisponde a tutta la gamma di tinte intermedie tra un colore fondamentale e l'altro (ad esempio viola 410 *nm*, blu 470 *nm*, giallo 580 *nm* , rosso 650 *nm*); allo stesso tempo, la luce bianca non corrisponde ad una specifica lunghezza d'onda, ma si

ottiene quando le oscillazioni, correlative alle lunghezze d'onda dell'intervallo visibile, colpiscono simultaneamente l'occhio umano, determinando appunto una fusione delle varie luci colorate che istituiscono lo spettro visibile: quanto detto può essere dimostrato facendo transitare un fascio di raggi solari attraverso un prisma di vetro (*figura 7*).

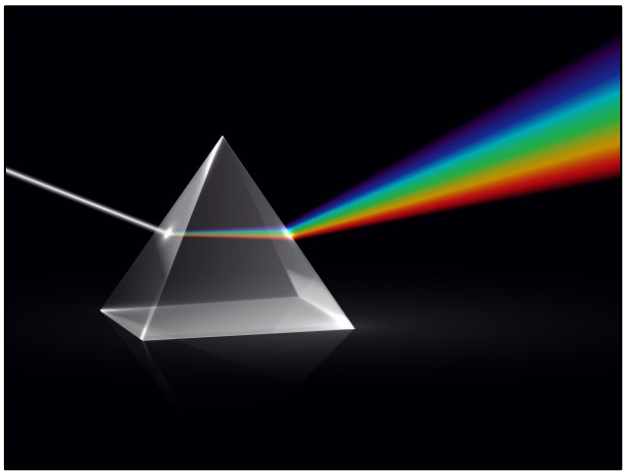

Figura 7 - Composizione della luce bianca

Inoltre, si può considerare, esaminando il grafico successivo (*figura 8*), l'andamento della sensibilità spettrale relativa dell'occhio umano:

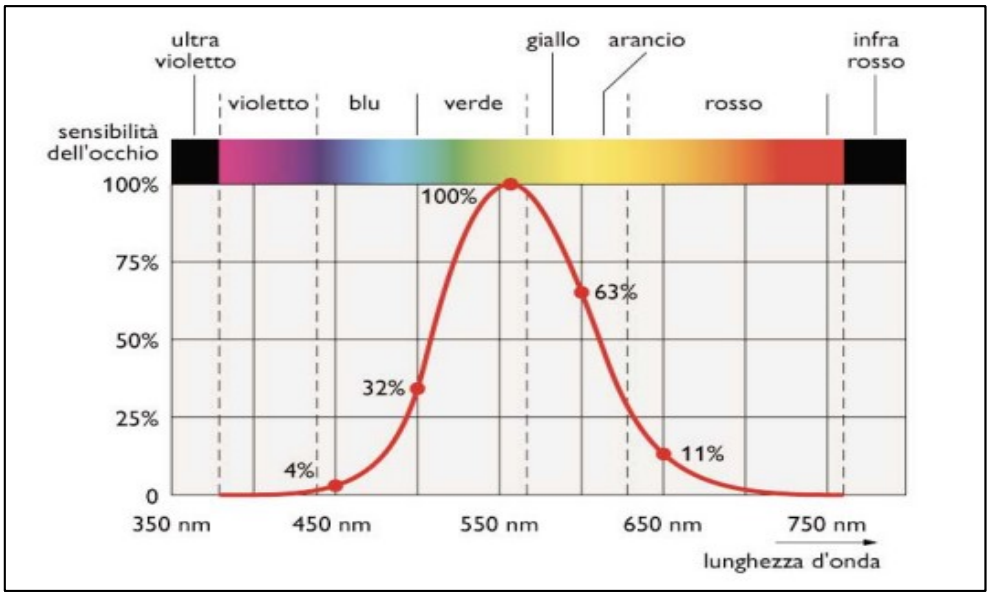

Figura 8 - Sensibilità spettrale dell'occhio umano

la sensibilità alla radiazione giallo-verde è considerata pari al 100%, a cui corrisponde un fattore di sensibilità visiva pari a uno, ed a quest'ultimo verranno messe a rapporto le sensibilità di tutte le altre lunghezze d'onda (ad esempio, il fattore di sensibilità dell'occhio per la radiazione di colore celeste è di 0,32).

2.1 Flusso della radiazione elettromagnetica

Risulta necessario soddisfare diverse esigenze indispensabili per poter conferire una corretta illuminazione ad un ambiente indoor, tra di esse ci sono il comfort visivo, la prestazione visiva e la sicurezza: per soddisfarle occorre esaminare alcuni parametri fondamentali dell'illuminotecnica, che contraddistinguono le analisi di un ambiente luminoso.

Viene considerata inizialmente la **radiazione elettromagnetica**, la cui prima formulazione è presentata in una serie di pubblicazioni del fisico britannico Maxwell; essa corrisponde all'energia nel campo elettromagnetico e può essere trattata secondo due punti di vista, che sono la *teoria ondulatoria* e la *teoria corpuscolare*. Secondo la teoria ondulatoria, una radiazione è costituita da un'onda elettrica e un'onda magnetica (radiazione elettromagnetica), che si propagano vibrando su piani ortogonali tra loro ed ortogonali rispetto alla direzione di propagazione (onde trasversali). Invece, secondo la teoria corpuscolare, la radiazione elettromagnetica è costituita da un fascio di particelle, dette fotoni, che si propagano in modo rettilineo, con moto sinusoidale di frequenza *v*. [9]

2.2 Flusso luminoso

Il **flusso luminoso** è una grandezza fotometrica che misura la quantità di luce emessa da una sorgente, in relazione alla variazione della sensibilità dell'occhio alle diverse frequenze della luce.

Si misura in *lumen* e viene definito come l'integrale, esteso nel campo di visibilità dell'occhio umano, del prodotto tra il ***fattore di visibilità relativa*** **V(λ)** e la ***radiazione emessa per ciascuna lunghezza d'onda*** **w(λ)**, espressa in Watt (dove λ è la lunghezza d'onda relativa i-esima): [9]

$$\Phi = 683 \int_{380\ nm}^{780\ nm} V(\lambda) w(\lambda) d\lambda \quad [lm] \tag{2.1}$$

Viene interposto all'integrale il fattore 683 $[\frac{lm}{W}]$ che corrisponde alla massima efficienza spettrale.

2.3 Intensità luminosa

L'**intensità luminosa** è definita come un flusso luminoso valutato in una certa direzione, a cui occorre associare un volume entro cui il flusso luminoso è contenuto, dato che il flusso stesso occupa uno spazio fisico tridimensionale. Il volume in questione è un cono (quindi un *angolo solido*) avente il vertice nella sorgente ed il cui asse individua la direzione; lo *steradiante* è l'unità di misura dell'angolo solido e, osservando la *figura 9*, si può constatare che l'***angolo in steradianti* Ω** è uguale al rapporto tra l'area **A** della calotta sferica (di raggio **R**) e l'area di un quadrato di lato pari al raggio:

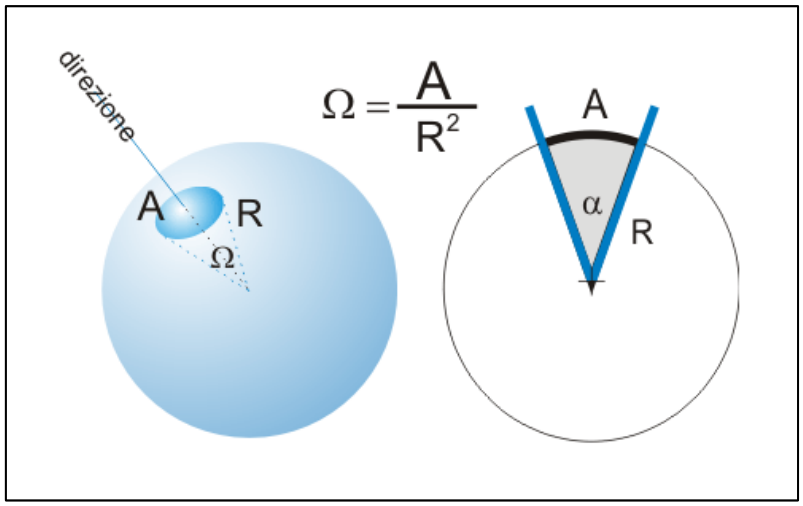

Figura 9 - Angolo in steradianti

Continuando ad esaminare la *figura 9*, è possibile definire in maniera più scrupolosa l'area A, notando che l'angolo solido può essere messo in relazione con l'angolo al vertice di una sezione assiale del cono (denominato α):

$$A=2\pi R^2\left(1-cos\frac{\alpha}{2}\right) \tag{2.2}$$

In definitiva, ponendo la relazione per la definizione dell'area A, appena citata, all'interno della relazione relativa all'angolo in steradianti **Ω**, si avrà:

$$\Omega=\frac{A}{R^2}=2\pi\left(1-cos\frac{\alpha}{2}\right) \qquad [str] \tag{2.3}$$

A questo punto, noti **dΩ**, ovvero l'*angolo solido infinitesimo che racchiude la direzione* e **dΦ**, ossia il *flusso contenuto nell'angolo solido,* si può definire la relazione dell'intensità luminosa come di seguito:

$$I = \frac{d\Phi}{d\Omega} \qquad [cd] \tag{2.4}$$

L'unità di misura è la *candela*, definita come l'intensità luminosa emessa, da una determinata direzione, da una sorgente che emette una radiazione monocromatica avente lunghezza d'onda pari a 555 *nm* e che ha, sempre in quella direzione, una potenza radiante specifica, quindi un'intensità radiante di $\frac{1}{683}W$ (il fattore 683, come visto nel paragrafo **2.2**, è stato assunto come fattore di visibilità nel punto di massimo). [9]

2.4 Luminanza

L'entità effettiva di un fascio luminoso che ha come sorgente una superficie, avente dimensioni estese, presa in considerazione nella direzione dell'osservatore della luce stessa, è espressa da una grandezza fotometrica chiamata **luminanza**. Si misura in *nit* e sperimentalmente viene definita come il rapporto tra l'intensità luminosa della sorgente, posta in direzione dell'osservatore, e la superficie posta nella maniera in cui viene percepita dall'osservatore stesso:

$$L = \frac{dI}{dS\cos\alpha} \qquad [nit]=[\frac{cd}{m^2}] \tag{2.5}$$

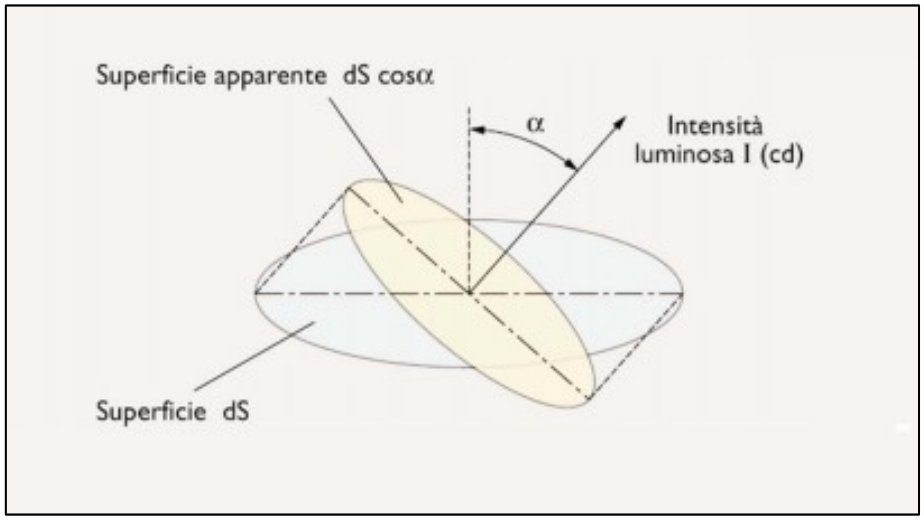

Figura 10 – Rappresentazione grafica della superficie apparente e dell'intensità luminosa

Nella relazione presentata, il termine ***dS*** corrisponde all'*area della sorgente*, mentre ***α*** è *l'angolo incluso tra l'osservatore e l'asse della superficie che effonde la luce.* [9]

2.5 Illuminamento

L'ultima grandezza fotometrica che verrà menzionata è l'**illuminamento**, il quale esprime l'entità di luce che investe una particolare superficie, senza considerare il punto di vista dell'osservatore. Si misura in *lux* ed è definito come il rapporto tra il flusso luminoso incidente su una superficie e la superficie stessa che viene presa in considerazione: [9]

$$E = \frac{d\Phi}{dS} \qquad [lux]=[\frac{lm}{m^2}] \tag{2.6}$$

La norma *UNI EN 12464-1* suggerisce di aumentare l'**illuminamento medio mantenuto** nel momento in cui il compito visivo sia critico, dovuto ad esempio ad una particolare esigenza di un grado di accuratezza o ad un'alta produttività richiesta, sia per tempi ridotti o per periodi prolungati, ma anche quando le capacità visive del lavoratore sono inferiori del normale. [7]

In conclusione, si può constatare che, per far in modo che una superficie venga investita dalla maggior parte della luce possibile, essa deve trovarsi in posizione perpendicolare con la fonte luminosa.

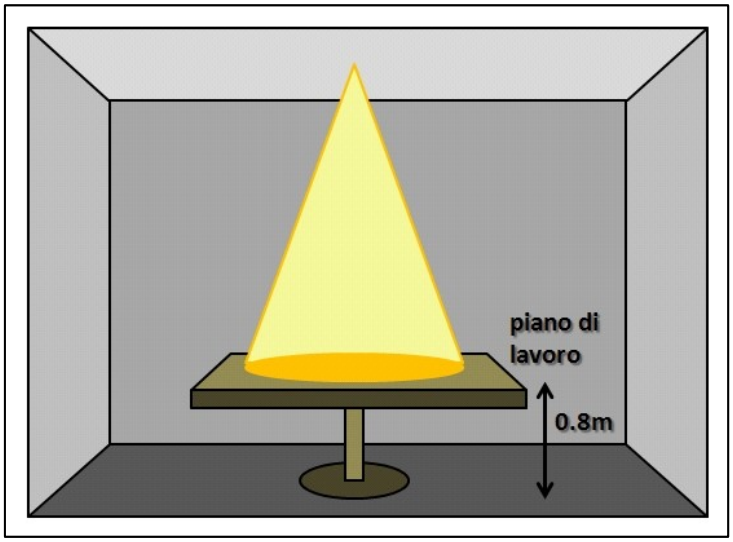

Figura 11 - Esempio di una fonte luminosa che investe un piano di lavoro

Capitolo 3

Sensore TSL2591 e relativi fotodiodi

All'interno del dispositivo per la misurazione delle variabili indoor (*Comfort Air*) è installato un sensore, ad alta sensibilità, per il rilevamento dell'intensità luminosa e per la sua conseguente conversione in output digitale, dove la trasformazione avviene con supporto per un'interfaccia diretta *I²C*, ovvero un sistema di comunicazione seriale bifilare, che garantisce una velocità di trasmissione dei dati fino a 400 *kbit/s*. Questo sensore, che prende il nome di ***TSL2591*** (*figura 12*), unisce due tipi di fotodiodo, uno a banda larga (visibile più infrarosso) ed uno di risposta, agli stessi infrarossi, in un circuito singolo integrato *CMOS* ("*Complementary Metal-Oxide Semiconductor*"); la scelta di utilizzare questo tipo di circuito è dovuto al fatto che uno dei principali benefici della logica *CMOS* è di avere una potenza statica dissipata idealmente nulla. [10]

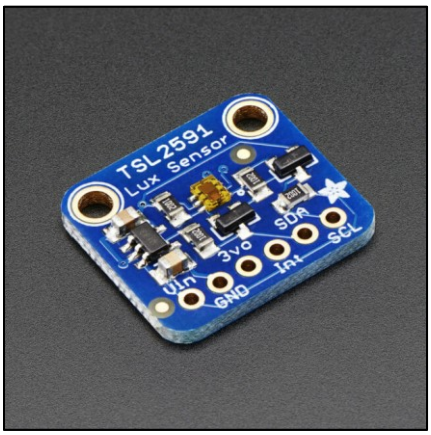

Figura 12 – Sensore TSL2591

3.1 Principio di funzionamento del *TSL2591*

Il sensore *TSL2591*, come detto, consente di misurare l'intensità luminosa di un ambiente, grazie ad una buona capacità di campionamento; in particolare, con un solo sensore, garantisce la rilevazione sia dello spettro *IR* (infrarossi) sia del campo visibile, con un range di misurazione compreso tra 188 *lux* e 88000 *lux.*

Nella figura seguente è possibile osservare il diagramma del circuito:

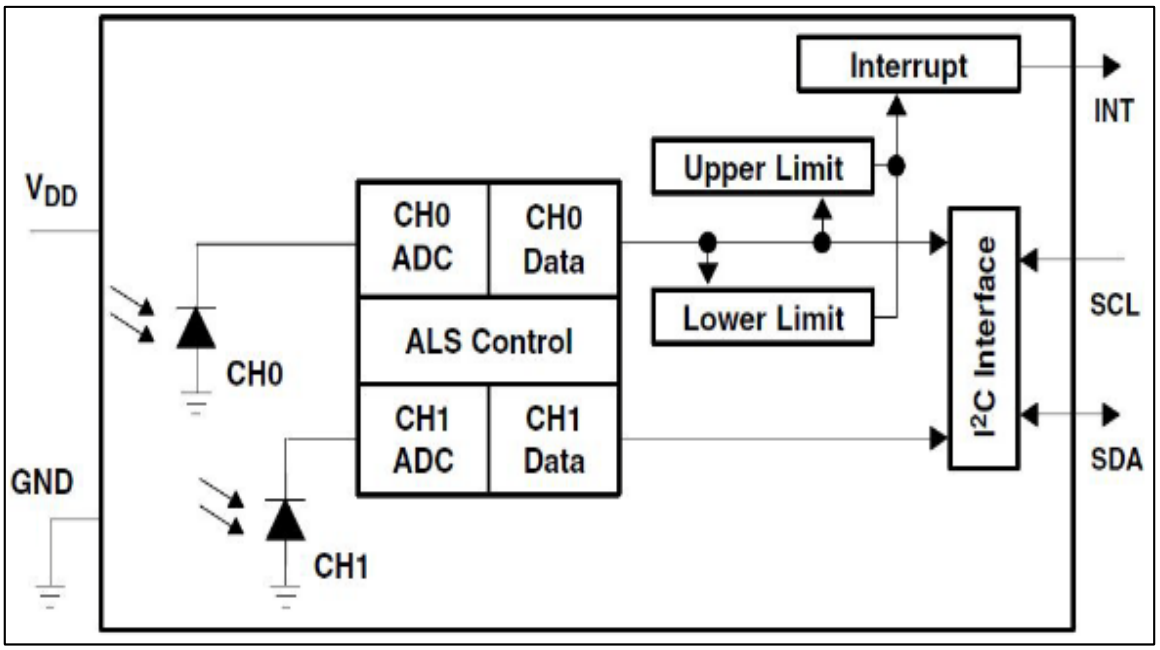

Figura 13 - Rappresentazione grafica del circuito del TSL2591

Le correnti del fotodiodo vengono trasformate, da due *ADC* ("*Analog to Digital Converter*") integrati, in un'uscita digitale, che riproduce l'irradianza, ovvero la densità di corrente termica trasmessa per irraggiamento, misurata su ogni canale.

L'uscita digitale del segnale può essere traslocata, in input, all'interno di un microprocessore, dove viene derivata la luminanza. Inoltre, il sensore *TSL2591* supporta un *interrupt* che rimane operativo fino a quando il firmware non lo riporta a zero.

Per quanto concerne la conversione dei due canali, essa viene eseguita contemporaneamente e, nel momento in cui il ciclo di conversione viene portato a termine, i dati vengono memorizzati nei relativi registri, chiamati ***CH0 Data*** e ***CH1 Data***. Successivamente alla avvenuta memorizzazione dei dati, viene compiuta la misurazione successiva, in maniera del tutto automatica.

Inoltre, come espresso nell'introduzione di questo capitolo, la comunicazione con il dispositivo è compiuta attraverso un'interfaccia bifilare standard I^2C diretta, ed è proprio attraverso quest'ultima che avviene la connessione tra il *TSL2591* ed il microcontrollore. Quanto detto si aggiunge al fatto che non sussiste l'esigenza di impiegare un circuito esterno per il condizionamento del segnale,

poiché il segnale stesso (in uscita) è digitale ed è adeguatamente esente al rumore durante il periodo in cui viene confrontato ad un segnale analogico.

Ulteriormente, il *TSL2591* possiede anche due livelli di soglia programmabili ("***Upper Limit***" e "***Lower Limit***") che originano un'interruzione laddove il valore della misurazione dell'intensità di luce oltrepassa questi limiti: una funzionalità notevole per ottimizzare il campionamento in base alle condizioni di luminosità dell'ambiente. Allo stesso tempo, se i valori dell'intensità di luce oltrepassassero i valori di soglia, per un periodo di tempo stabilito, sarebbe possibile comandare un *interrupt* al segnale per determinate soglie del segnale, al contrario di quanto visto precedentemente, dove il segnale veniva interrotto immediatamente. [11]

3.2 Fotodiodi

Il **fotodiodo** è un componente che ha il compito di convertire un segnale ottico in ingresso, quale la luce incidente, in un segnale elettrico in uscita, come la corrente elettrica.

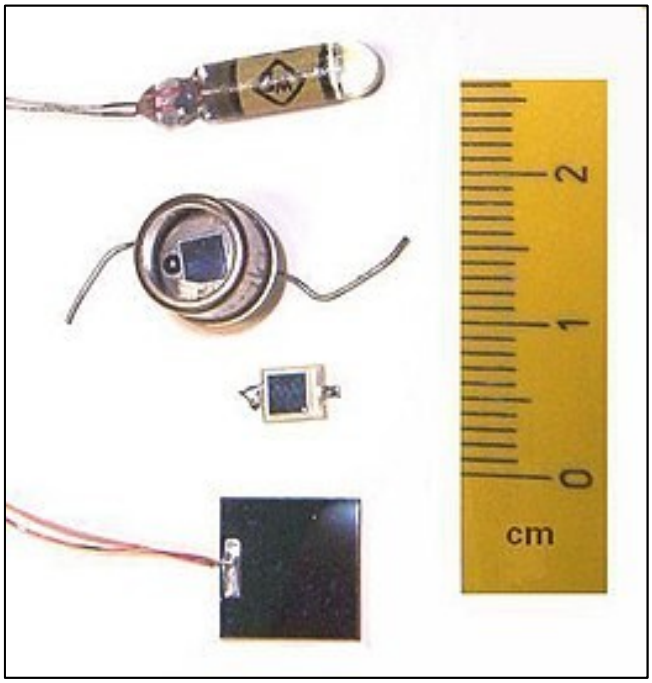

Figura 14 - Alcune tipologie di fotodiodi

Nella *figura 14* vengono mostrati diversi tipi di fotodiodi: a tal proposito è bene ricordare che la diversa tipologia di materiale, con cui vengono realizzati, assume un'importanza critica per poter

garantire il loro regolare funzionamento, infatti, da essi scaturisce l'energia minima di cui il fotone dovrà essere dotato, per poter generare la fotocorrente.

Entrando più nel particolare, i materiali più comunemente utilizzati per la produzione dei fotodiodi sono riportati nella tabella seguente:

Tabella 3 - Materiali utilizzati per i fotodiodi

MATERIALE	LUNGHEZZA D'ONDA [*nm*]
Silicio	190-1100
Germanio	800-1700
Arseniuro di indio gallio	800-2600
Solfuro di piombo	980-3500

Infine, al fine di ottenere le massime prestazioni, è bene prestare particolare attenzione ai circuiti di interfaccia, i quali devono essere appropriati per ottenere la corrente massima, al mutare di certe variabili, come ad esempio l'intensità luminosa, la lunghezza d'onda e l'allineamento ottico-meccanico. [12]

3.2.1 Giunzione p-n

I fotodiodi si basano su **giunzioni *p-n***, cioè costituite da una zona con un eccesso di lacune (strato *p*) e una zona con eccedenza di elettroni (strato *n*), permettendo alla corrente elettrica di fluire unidirezionalmente.

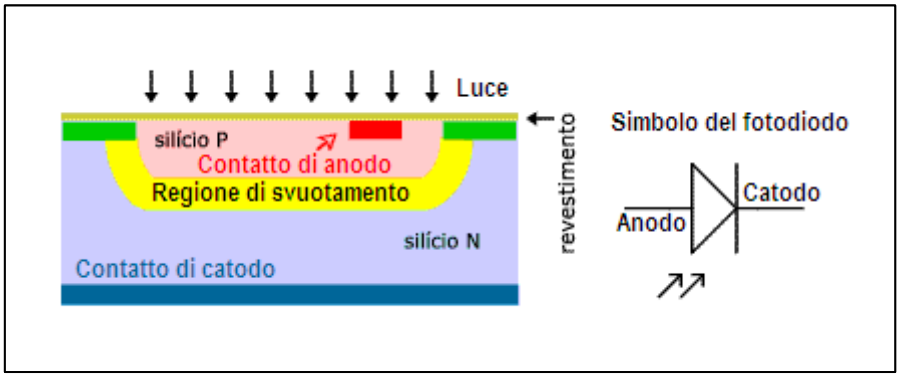

Figura 15 - Sezione trasversale e simbolo del fotodiodo

Nella *figura 15* viene mostrata la sezione trasversale di un fotodiodo: il materiale di strato *p* è la superficie attiva ed il materiale di strato *n* forma un substrato di una giunzione *p-n* che funziona come

un convertitore fotoelettrico. Si può constatare che, in questo tipo di giunzione, si configura una **regione di svuotamento** ("*depletion layer*"), cioè uno strato sottile neutro, in cui un drogaggio (ovvero un'introduzione, in quantità controllate, di atomi estranei in un composto puro, generalmente un cristallo, al fine di modificarne le proprietà) di tipo *p* si approssima ad un drogaggio di tipo *n*, senza alcun contatto. Quando sono soggetti a drogaggio, i semiconduttori, sia di tipo *n* sia di tipo *p*, sono conduttori tanto migliori quanto più elevato è il drogaggio stesso, mentre la regione di svuotamento detiene le proprietà di un isolante.

Nella regione di svuotamento (detta anche **zona di carica spaziale**), le particelle portatrici di carica, rispettivamente del lato *p* e del lato *n*, dinnanzi ad un gradiente avente entità elevata, causato dal differente tipo di drogaggio, propagano nel semiconduttore adiacente e nel frattempo generano una corrente di diffusione, facendo rimanere non compensati gli atomi ionizzati dei droganti (ovvero soggetti a drogaggio); gli stessi atomi ionizzati, a loro volta, origineranno una *differenza di potenziale* (in particolare, si crea ai bordi della regione di carica spaziale e viene anche chiamata **tensione di "*built-in*"**), un *campo elettrico* e, nell'istante in cui vengono spostati i portatori, una *corrente di trascinamento*, opposta alla corrente di diffusione. La larghezza correlata alla zona di carica spaziale dipende dai drogaggi e da ciascun lato (che sia *p* o che sia *n*) è inversamente proporzionale al drogaggio del semiconduttore. [13]

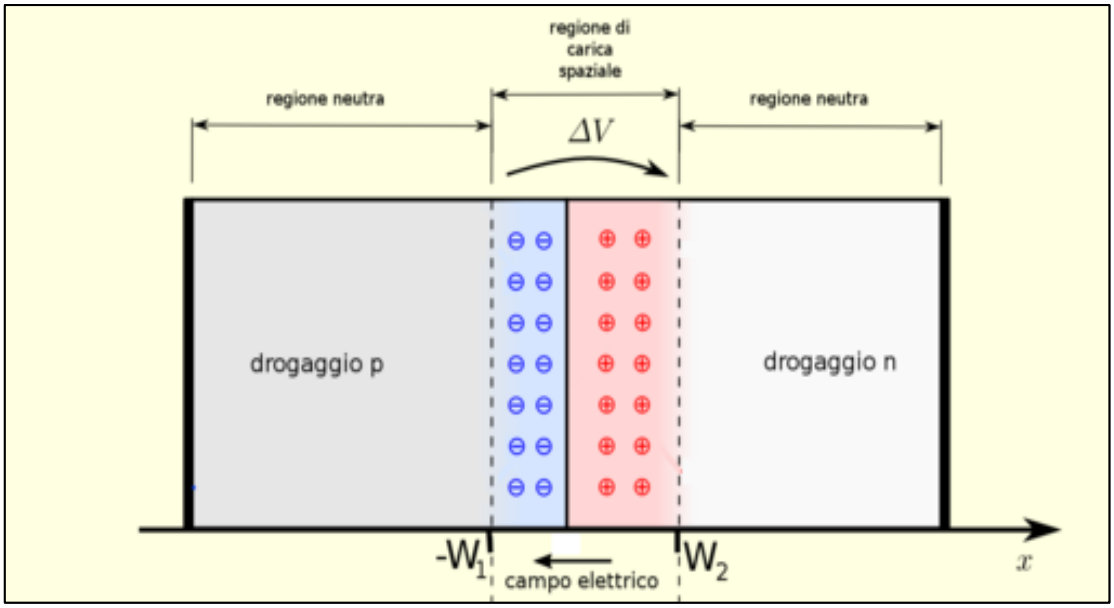

Figura 16 - Rappresentazione della giunzione p-n in un fotodiodo

3.2.2 Polarizzazione diretta ed inversa

La giunzione p-n può essere impiegata come un diodo grazie alle sue proprietà di conduzione in regime di **polarizzazione diretta** e **polarizzazione inversa**.

Analizzando il primo caso, si ha polarizzazione diretta quando al terminale positivo del generatore di tensione è collegata la parte di tipo *p*, ed al contempo al terminale negativo è connessa la parte di tipo *n*. In questo caso, le lacune nella regione di tipo *p* e gli elettroni nella regione di tipo *n* sono spostati verso la giunzione: ciò comporta alla riduzione dell'ampiezza della regione svuotata e, poiché la tensione positiva applicata si concentra quasi completamente ai capi della regione di svuotamento, la differenza di potenziale sulla giunzione decresce. [13][14]

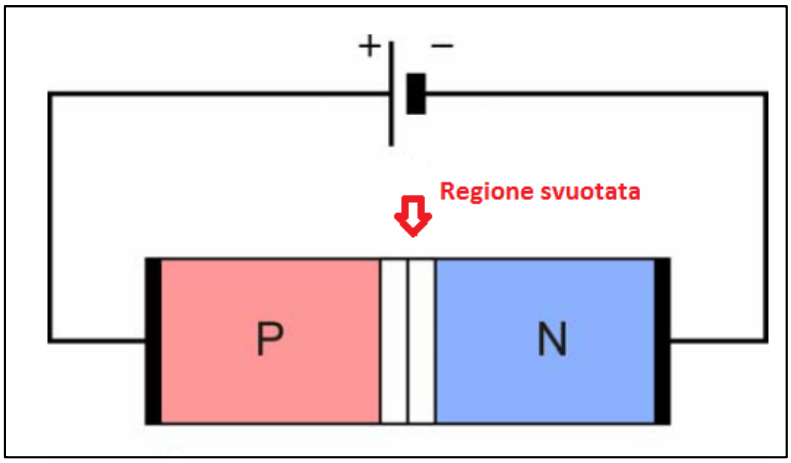

Figura 17 - Polarizzazione diretta

Il secondo caso prende in considerazione una polarizzazione inversa; contrariamente al caso precedente, in questa circostanza viene collegata la regione di tipo *p* al terminale negativo del generatore di tensione mentre viene connessa la regione di tipo *n* al terminale positivo. Visto che al terminale negativo è collegata la regione di tipo *p*, le lacune inerenti a questa zona vengono distanziate dalla giunzione ed in questo modo l'ampiezza della zona svuotata subirà un aumento. Stesso ragionamento per gli elettroni appartenenti alla regione di tipo *n*, che vengono allontanati dalla giunzione a causa del comportamento del terminale positivo del generatore di tensione. Queste considerazioni implicano che ai capi della regione di svuotamento si raccolga la tensione negativa applicata al dispositivo e ciò comporta ad un aumento della barriera di potenziale. [13][14]

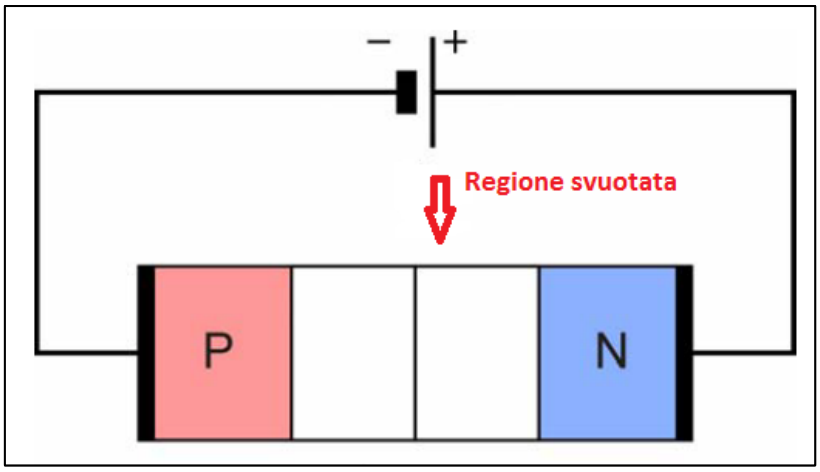

Figura 18 - Polarizzazione inversa

Dopo aver analizzato i casi di polarizzazione diretta e inversa, viene descritta, dall'*equazione di Shockley*, la **caratteristica ideale della giunzione p-n**, data da: [14]

$$i_D = i_S(e^{\frac{V_D}{nV}} - 1) \qquad [A] \qquad (3.1)$$

dove:

$\boldsymbol{i_D}$ = intensità di corrente sul diodo;

$\boldsymbol{i_S}$ = intensità di corrente di saturazione;

$\boldsymbol{V_D}$ = differenza di potenziale tra i due terminali del diodo;

$\boldsymbol{n}$ = parametro adimensionale costruttivo del cristallo;

$\boldsymbol{V}$ = tensione termica.

3.2.2 Supporto per il fotodiodo

La zona *p* inerente al fotodiodo, disposta molto vicino alla sua struttura esterna, è ricoperta da uno strato antiriflesso, sul quale, a sua volta, è posta una lente il cui obiettivo è quello di conferire un andamento perpendicolare ai raggi luminosi incidenti sulla superficie. Perciò, per poter mantenere stabile la lente e per non far variare la sua posizione rispetto al fotodiodo, è stato posizionato un supporto realizzato in *PLA* (Acido Polilattico);

questa plastica biodegradabile (il *PLA*) ha ottime caratteristiche di resistenza e, soprattutto, è semplice da stampare, infatti è adatta ad ogni tipo di stampante 3D. Così facendo, inserendo questo supporto, la luce raggiunge il dispositivo sempre allo stesso modo ed è quindi possibile applicare, per prove consecutive, la stessa curva di taratura.

Nella *figura 19* è raffigurato il disegno del piccolo supporto, realizzato con il software "*Ultimaker Cura*":

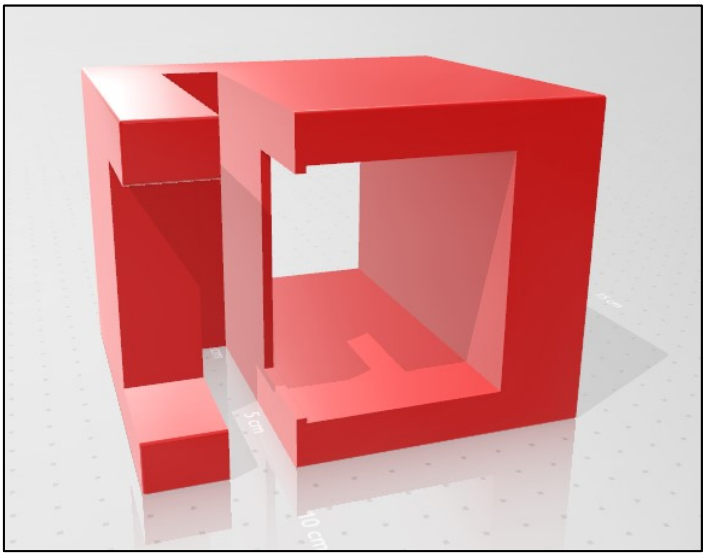

Figura 19 - Supporto in PLA per il fotodiodo

Capitolo 4

Calibrazione ed analisi dei dati

Allo scopo di effettuare la calibrazione del sensore di luminosità, installato all'interno del dispositivo per la misurazione delle variabili indoor (*Comfort Air*), è risultata necessaria la realizzazione di un *set-up*, ovvero un banco di prova, che potesse adempiere tutti i requisiti richiesti per una corretta e precisa fase di acquisizione dei dati. Il sensore di riferimento impiegato è il medesimo rispetto a quello installato all'interno del dispositivo *Comfort Air*: si tratta del sensore *TSL2591*, descritto nel capitolo **3.1**, un sensore per la misurazione, in ambienti indoor, dell'intensità luminosa. Successivamente all'acquisizione dei dati, avvenuta utilizzando anche i software "*Putty*" e "*WinSCP*", vi è stata l'analisi dei dati stessi, ottenuti dalle rilevazioni eseguite, ed in seguito la calibrazione del sensore tramite il software di programmazione "*Python*"; ultimata anche questa fase, sono nate le dovute considerazioni finali, tenendo conto anche del livello di incertezza del sensore esaminato.

Figura 20 - Posizionamento del sensore di riferimento TSL2591 sul Comfort Air

4.1 Set Up

Come detto precedentemente, per la procedura di acquisizione dei dati, è stato necessario creare un *set-up*, nel quale sono stati utilizzati i seguenti dispositivi:

- *Raspberry Pi 0*;
- dispositivo per la misurazione delle variabili indoor (*Comfort Air*);
- sensore di riferimento *TSL2591*;
- faretto alogeno;
- regolatore di tensione (*dimmer*);
- router *Wi-Fi*;
- laptop.

Figura 21 - Dispositivi utilizzati per l'acquisizione dei dati (da sinistra Comfort Air, sensore di riferimento TSL2591, Raspberry Pi 0, laptop, router, faretto alogeno, dimmer)

4.1.1 Raspberry Pi 0

Il sensore di riferimento *TSL2591* è stato collegato ad un dispositivo chiamato *Raspberry Pi 0* (*figura 21*): esso è a tutti gli effetti un computer programmabile e si caratterizza per il fatto che assegna un set-up dell'hardware già predisposto per l'utilizzo, una caratteristica che agevola il processo di assemblaggio della base tecnica per i progetti che vengono costituiti. Difatti, il *Raspberry Pi 0* dispone già di tutti i componenti necessari per far funzionare il minicomputer autonomamente, così da poter essere utilizzato come piattaforma di programmazione, resa veloce ed intuitiva impiegando un linguaggio di scripting evoluto e moderno come *Python*. [15]

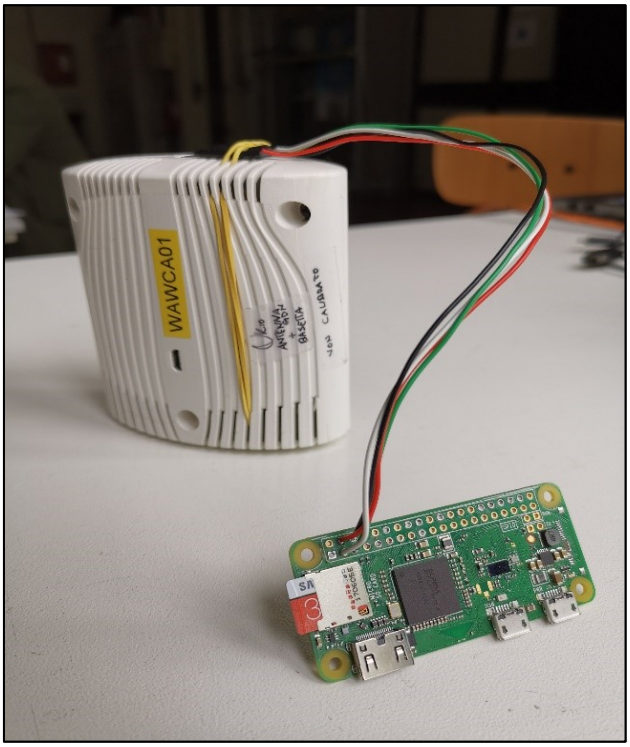

Figura 22 - Raspberry Pi 0 connesso al TSL2591, situato sulla superificie superiore del Comfort Air

Prima di andare a definire in dettaglio questi parametri, occorre introdurre un fattore fondamentale per il calcolo del *LUX*, ovvero il ***cpl***, un parametro che dipende dal tempo di acquisizione, dal gain (nella relazione, '*again*' è semplicemente una funzione del gain) e da un parametro fornito dal produttore inerente al datasheet (documentazione) del sensore *TSL2591*:

$$cpl = \frac{atime * again}{TSL2591_{LUX_{DF}}} \tag{4.1}$$

A questo punto , si possono analizzare i tre parametri di *LUX*, a partire da ***LUX 1***, un termine che tiene maggiormente in considerazione i valori in entrata nel canale *CH0*, che corrispondono al campo visibile più infrarosso. [16]

Allo stesso tempo, a causa dell'andamento altalenante dei valori in entrata nel canale *CH1*, i valori ottenuti da quest'ultimo vengono moltiplicati per un coefficiente correttivo (minore di 1, inserito nei datasheet del sensore *TSL2591*), che comporta ad una minimizzazione della variazione del valore complessivo *LUX1*. [16]

Perciò la relazione del *LUX1* è la seguente:

$$LUX1 = \frac{[(float)CH0 - (float)CH1 * TSL2591_{LUX_{COEFB}}]}{cpl} \tag{4.2}$$

Il parametro ***LUX 2*** invece, a causa del fluttuare dei valori, tiene conto di un fattore correttivo sia per quanto concerne il canale *CH0* sia per il canale *CH1* (entrambi i coefficienti correttivi sono prelevati dai datasheet del sensore *TSL2591*). Per questo il *LUX2* viene considerato come il parametro che rappresenta, più fedelmente possibile, il campo visibile poiché, come detto in questo caso, la componente dell'infrarosso viene compensata da un coefficiente moltiplicativo, come evidente dalla seguente relazione:

$$LUX2 = \frac{[(float)CHO * TSL2591_{LUX_{COEFC}} - (float)CH1 * TSL2591_{LUX_{COEFD}}]}{cpl} \tag{4.3}$$

In questo caso, il coefficiente correttivo moltiplicato per *CH1* ($TSL2591_{LUX_{COEFD}}$) è diverso rispetto a quello usato precedentemente per il calcolo del *LUX1* ($TSL2591_{LUX_{COEFB}}$) poiché tiene in considerazione anche del coefficiente moltiplicativo utilizzato per *CH0*. [16]

Infine, una definizione alternativa rispetto alle due precedenti è data dal parametro ***LUX***, definito come valore mediato tra il parametro *LUX1* e il *LUX2*, espresso dalla relazione seguente:

$$LUX = \frac{[(float)CH0 - (1.7 * (float)CH1)]}{cpl} \tag{4.4}$$

In questo caso, non è presente un fattore correttivo prelevato dai datasheet del sensore *TSL2591*, ma, per il canale *CH1*, viene moltiplicato un fattore numerico decimale. [16]

4.1.2 Dimmer (regolatore di tensione)

Nota la necessità di garantire un range di lux abbastanza ampio per le acquisizioni, è stato implementato, al faretto alogeno, un regolatore di tensione, chiamato "***dimmer***" (*figura 23*), attraverso il quale è stato possibile esaminare con attenzione e quindi controllare il voltaggio in entrata per poter lavorare in un intervallo di *lux* opportuno per le analisi svolte.

Figura 23 - Dimmer

Il suo funzionamento è semplice ed intuitivo dato che la regolazione dell'intensità luminosa, emessa dal faretto, è eseguibile manualmente. Infatti, all'interno del dispositivo troviamo un circuito che ha delle resistenze variabili che, a loro volta, in base alla rotazione della manopola, conferiscono la regolazione della tensione in uscita.

Il *dimmer* utilizzato ha una potenza massima di 3 *kW* ed è alimentato con una tensione in ingresso di 220 *V*. La scelta del faretto da 900 *W* è stata fatta affinché l'emissione dei lux fosse sufficiente per la calibrazione del sensore, ed in particolar modo fino al valore di 700 *lux* circa, dato che negli ambienti indoor difficilmente vengono superati.

Nota che la potenza del *dimmer* fosse superiore a quella del faretto alogeno, la sensibilità della regolazione dell'intensità luminosa è stata più accurata, dato che ad una leggera variazione della tensione in entrata è coincisa un lieve variazione di emissione luminosa da parte del faretto.

4.1.3 Laptop e Router Wi-Fi

Oltre a tutti i dispositivi già descritti, è stato necessario creare una rete *Wi-Fi* per far in modo che i sensori all'interno del *Comfort Air* e il *TSL2591* potessero connettersi ed allo stesso tempo indirizzare i dati acquisiti dalle rilevazioni con un laptop, in modo da poterli successivamente memorizzare ed analizzare.

Sia il *Comfort Air*, sia il dispositivo *Raspberry Pi 0* montano una scheda *Wi-Fi* per la connessione e l'invio successivo dei dati acquisiti.

4.2 Acquisizione dei dati

Per l'acquisizione dei dati è stata scelta una stanza che registrasse, per quanto possibile, un valore di *lux* nullo, dunque un luogo buio, privo di finestre e di fonti luminose artificiali provenienti da sorgenti diverse dal faretto alogeno utilizzato durante la sperimentazione, poiché diversamente avrebbero potuto interferire con i valori ottenuti dalle acquisizioni effettuate.

Il *Comfort Air* ed il sensore di intensità luminosa di riferimento *TSL2591* sono stati posizionati su una scrivania ad uno stesso livello di acquisizione poiché è noto che l'illuminamento dipende anche dalla distanza della sorgente luminosa rispetto al sensore che ne preleva l'intensità; in particolare, il faretto alogeno è stato collocato a circa 30 *cm* di distanza ed alimentato da una corrente di 220 *V* (come già detto nel paragrafo **4.1.2**) tramite il regolatore di tensione (*dimmer*).

Formulato, con il software *Python*, il codice per l'ottenimento dei dati (riportato in ***Appendice A***), durante la fase di acquisizione sono stati rilevati, per il *Comfort Air*, i dati relativi ai sensori di temperatura, umidità relativa, intensità luminosa, anidride carbonica e composti organici volatili mentre per il sensore di riferimento i tre parametri di *LUX* (descritti nel paragrafo **4.1.1**), ovvero *LUX1*, *LUX2* e *LUX*.

Successivamente sono stati utilizzati i software "*Putty*" e "*WinSCP*" rispettivamente per la fase di comunicazione con il dispositivo *Raspberry Pi 0* e per la rilevazione dei dati acquisiti dai sensori inerenti al *Comfort Air* e dal sensore di riferimento *TSL2591*.

Durante la fase di acquisizione sono stati creati due file testuali (*.txt*), denominati "***raw_data1***" e "***raw_data2***", nella quale al loro interno sono stati inseriti in colonna tutti i dati che i sensori registravano. In particolare, nel file testuale "*raw_data1*" venivano salvati i valori inerenti ai sensori installati all'interno del Comfort Air mentre nel file testuale "*raw_data2*" sono stati inseriti i valori

del sensore di riferimento *TSL2591*. Nella figura seguente è mostrato un esempio di come i dati acquisiti dai sensori venivano salvati nei due diversi file testuali:

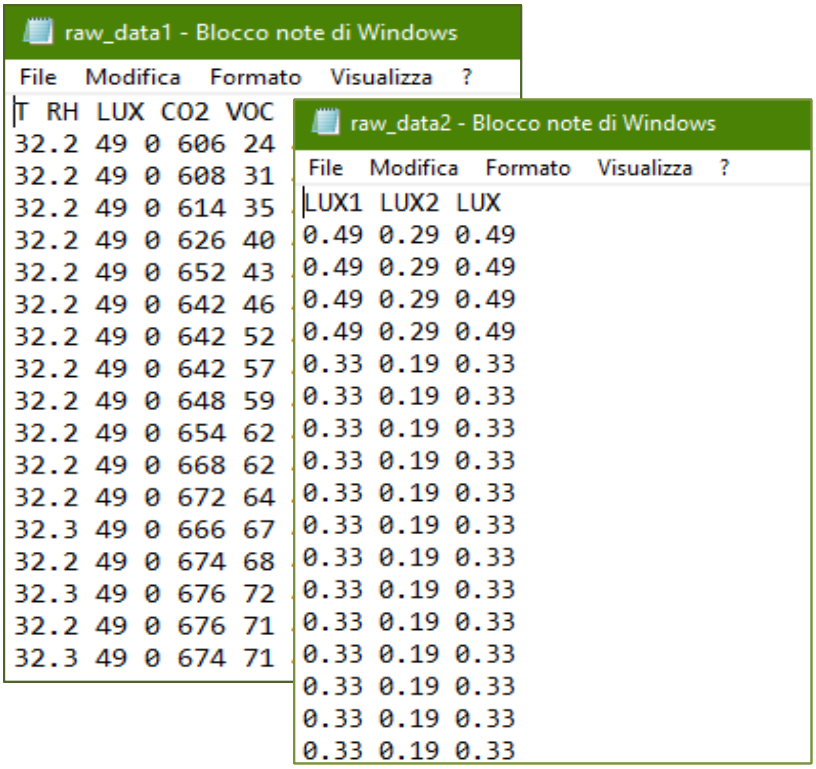

Figura 24 - File testuali "raw_data1" e "raw_data2"

Con l'utilizzo del dispositivo *dimmer*, connesso al faretto alogeno, sono stati eseguiti quattro step di regolazione dell'intensità luminosa, la quale incideva sulla lente del *Comfort Ai*r e sul sensore di riferimento *TSL2591*:

- 0 *lux* (faretto spento);
- ca. 100 *lux*;
- ca. 500 *lux*;
- ca. 700 *lux* (massima emissione possibile, poiché in ambienti indoor questo parametro non viene quasi mai oltrepassato [*cfr.* Tabella 2 - Valori lux consigliati in ambienti indoor]).

Per ognuno dei quattro step di regolazione dell'intensità luminosa sono state eseguite rispettivamente 40 acquisizioni (per un totale di 160 acquisizioni) sia per i dati inerenti ai sensori installati all'interno del dispositivo per la misurazione delle variabili indoor sia per i dati inerenti al sensore di riferimento *TSL2591*.

4.3 Importazione dei dati

Dopo aver raccolto tutti i dati necessari per la calibrazione del sensore di luminosità all'interno del *Comfort Air*, è stato necessario processarli. Anche in questo caso è stato utilizzato il software *Python* per formulare un codice (riportato in ***Appendice B***) che, in questo caso, potesse analizzare i dati ottenuti ma soprattutto che mettesse in relazione i valori acquisiti dal sensore di riferimento *TSL2591* con quelli ottenuti dal sensore di luminosità all'interno del *Comfort Air* ed arrivare infine al calcolo della sua incertezza.

A tal proposito, sono stati eseguiti diversi step per giungere alla calibrazione del sensore ed al calcolo della sua incertezza, sintetizzati di seguito ed approfonditi durante il corso di questo paragrafo:

1) importazione dei file testuali "*raw_data*" e conseguente creazione di vettori contenenti i valori acquisiti, sia per quanto concerne il *Comfort Air* sia per il sensore *TSL2591*;
2) utilizzo dei vettori per l'analisi dell'andamento dei valori ottenuti per la calibrazione;
3) utilizzo della "*regression line"* per trovare la retta di taratura, dato che l'andamento dei valori è stato osservato essere lineare;
4) calcolo dell'incertezza del sensore di luminosità all'interno del Comfort Air tramite l'utilizzo dello *scarto quadratico medio RMSE* ("*Root-Mean-Square-Error"*) e del coefficiente *R-squared* per il calcolo dell'incertezza del sensore.

I dati ottenuti dalle acquisizioni sono stati quindi importati nella shell interattiva di *Python* per poter essere processati e successivamente analizzati. Come riportato nel paragrafo **4.1.1**, rispetto al sensore di riferimento, è noto che per il sensore *TSL2591* sono tre le tipologie di parametri *LUX* di riferimento utilizzabili, quindi è stato arricchito il codice utilizzato su *Python* in modo tale che il software suddividesse i tre diversi parametri in tre diversi vettori.
Invece, per il *Comfort Air* è stato necessario prendere in considerazione solo i valori ottenuti dal sensore di luminosità, installato al suo interno, per creare un vettore, con il software *Python*, che contenesse i valori in *lux* ed al contempo che potesse scartare, poiché non necessari ad essere esaminati, tutti i restanti dati acquisiti dagli altri sensori installati all'interno del *Comfort Air*.

Un'osservazione da citare riguarda la procedura di acquisizione: durante questa fase si è notato che ogni qualvolta il programma *WinSCP* veniva avviato, come primo valore del *Comfort Air* veniva registrato un valore sistematicamente errato; per risolvere tale problema, è stata implementata un'ulteriore funzione nel codice per poter eliminare ogni primo valore che veniva salvato all'avvio di ogni step di acquisizione.

I vettori di riferimento corrispondenti ai valori *LUX* del sensore di riferimento *TSL2591* sono stati quindi messi in relazione con il vettore inerente ai valori *lux* del sensore di luminosità da calibrare all'interno del *Comfort Air*.
Sebbene ogni parametro di riferimento sia stato relazionato con i valori da calibrare, è stato scelto il parametro *LUX 2* come valore di riferimento perché questo valore è quello che più si avvicina alla percezione dell'occhio umano, utilizzando fattori correttivi sia per quanto riguarda la banda larga sia per l'infrarosso.

Iniziando lo studio dei dati è opportuno considerare il processo di **taratura statica** poiché tutte le caratteristiche riguardanti le prestazioni statiche dipendono, in un modo o in un altro, da questo processo.

In generale, la taratura statica si riferisce a situazioni in cui tutti gli ingressi (desiderati, interferenti, modificanti), eccetto uno, sono fissati a valori costanti. Poi l'ingresso sotto osservazione viene fatto variare su un certo insieme di valori costanti e di conseguenza anche le uscite variano con valori costanti all'interno di un certo campo. Le relazioni tra ingresso e uscita sviluppate in questo modo costituiscono una taratura statica valida sotto le fissate condizioni di costanza di tutti gli altri ingressi. [17]

Come si procede per una taratura statica, i valori *LUX 2* del sensore di riferimento *TSL2591* sono stati utilizzati come variabile q_o mentre i valori *lux* del sensore di luminosità all'interno del *Comfort Air* come variabile q_i.

Lavorando ancora con il software *Python*, sono state inserite graficamente le variabili e si è notato come i valori seguissero un andamento pressoché lineare fino al valore desiderato per la calibrazione, come si può constatare dalla *figura 25*:

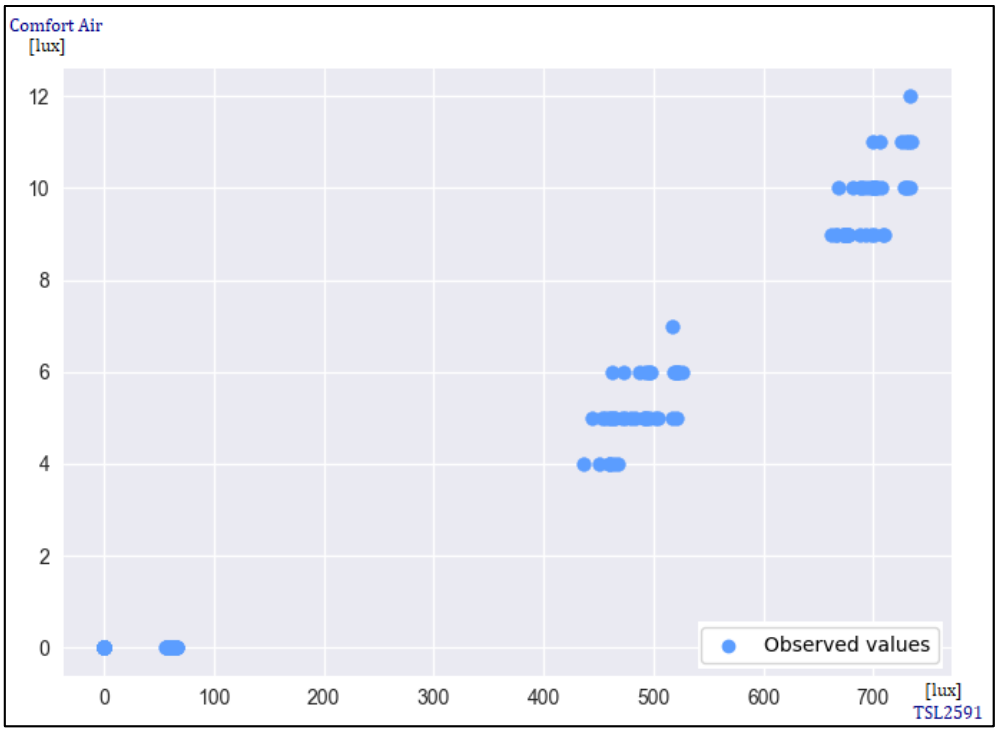

Figura 25 - Andamento dei valori lux del sensore da calibrare rispetto al sensore di riferimento

4.4 Calibrazione del sensore

Giunti a questo punto, l'obiettivo è stato quello di andare a calibrare il sensore di luminosità all'interno del *Comfort Air*.

Partendo dalle analisi sull'andamento dei valori ottenuti, risultato essere lineare, è stata realizzata, con *Python*, la retta dei minimi quadrati, anche definita "***regression line***". Per ciò che concerne l'operazione di taratura, il valore viene variato a passi discreti all'interno di un determinato intervallo e di conseguenza la lettura del valore di riferimento misurato varierà all'interno di un determinato intervallo.

La procedura richiede soltanto che l'intervallo scelto venga coperto sia per valori crescenti, sia per valori decrescenti; così un certo valore di riferimento viene applicato al massimo due volte, nel caso in cui si scelga di prendere lo stesso set di valori di riferimento in ingresso sia in direzione crescente, sia in direzione decrescente. [17]

4.4.1 Retta dei minimi quadrati (regression line)

La *curva media di taratura* per uno strumento di tale tipo viene generalmente assunta come la linea retta che, secondo certi criteri scelti, interpola al meglio i punti dispersi. Il criterio più usato è quello dei minimi quadrati, che sostanzialmente minimizza la somma dei quadrati delle differenze, nella direzione verticale, tra i punti ottenuti dai valori analizzati e la linea. [17]

L'equazione considerata per tale retta, chiamata "***regression line***", è:

$$q_o = mq_i + b \tag{4.5}$$

dove:

- $\boldsymbol{q_o}$ è la grandezza in uscita (variabile dipendente);
- $\boldsymbol{q_i}$ è la grandezza in ingresso (variabile indipendente);
- $\boldsymbol{m}$ è il coefficiente angolare della retta;
- $\boldsymbol{b}$ è il punto d'intersezione tra la retta e l'asse verticale (intercetta all'origine).

Le equazioni, implementate con le opportune funzioni di *Python*, utilizzate per calcolare il coefficiente angolare della retta ed il punto d'intersezione tra la retta e l'asse verticale, sono:

$$m = \frac{N\sum q_i q_o - (\sum q_i)(\sum q_o)}{N\sum {q_i}^2 - (\sum q_i)^2} \tag{4.6}$$

$$b = \frac{(\sum q_o)(\sum q_i)^2 - (\sum q_i q_o)(\sum q_i)}{N\sum {q_i}^2 - (\sum q_i)^2} \tag{4.7}$$

dove $\boldsymbol{N}$ è il numero totale dei punti considerati. [18]

L'equazione della *regression line* ottenuta, utilizzando il codice formulato con *Python*, per il caso esaminato in questa disamina, è:

$$y = 0{,}01x - 0{,}57 \tag{4.8}$$

Il risultato conseguito è osservabile anche graficamente tramite la figura seguente:

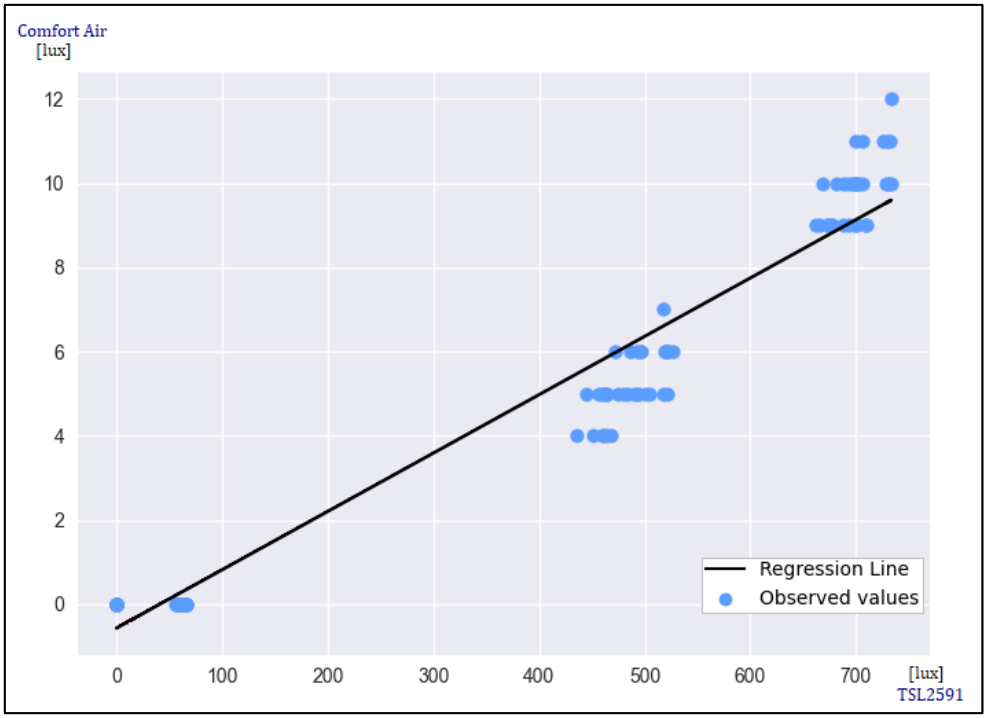

Figura 26 - Linea di regressione ottenuta con i valori lux del Comfort Air e del TSL2591

4.4.2 R-squared e RMSE

Una taratura come quella di *figura 26* permette la scomposizione dell'errore totale di un processo di misura in due componenti, la *parte sistematica* e la *parte casuale*. Sicuramente, una volta che lo strumento è stato tarato, l'errore sistematico può essere annullato proprio grazie alla taratura e l'unico errore rimanente è quello casuale, diverso per ogni lettura, non eliminabile ed al più contenuto entro un intervallo limitato che rappresenta l'incertezza strumentale. [17]

A tal proposito, quando si costruisce un modello di regressione lineare, è particolarmente importante comprendere quanto sia efficiente la sua capacità predittiva, ovvero quanto le variabili indipendenti riescono a intuire correttamente i valori della variabile dipendente.

Perciò, a partire dalla retta di taratura, è stato calcolato, tramite il software *Python*, il **coefficiente di determinazione *R-squared*** (o più comunemente $\boldsymbol{R^2}$), un indicatore che si basa sulla proporzionalità tra la variabilità dei dati e la correttezza del modello statistico utilizzato, e che quindi misura la frazione della varianza della variabile dipendente espressa dalla retta di regressione. [19]

La relazione per il calcolo di questo coefficiente è la seguente:

$$R^2 = \frac{ESS}{TSS} = 1 - \frac{RSS}{TSS} \tag{4.9}$$

dove:

- $ESS = \sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2$ (devianza spiegata dal modello) (4.10)
- $TSS = \sum_{i=1}^{n}(y_i - \bar{y})^2$ (devianza totale) (4.11)
- $RSS = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$ (devianza residua) (4.12)

Nelle formule 4.10, 4.11 e 4.12, $\boldsymbol{y_i}$ sono i dati osservati, $\boldsymbol{\bar{y}}$ la media dei dati osservati e $\boldsymbol{\hat{y}_i}$ sono i dati stimati dal modello ottenuto dalla regressione.

Questo coefficiente può assumere valori compresi fra 0 e 1: se sono prossimi ad uno significa che i regressori predicono bene il valore della variabile dipendente in campione, mentre se è pari a zero significa che non lo fanno. [19]

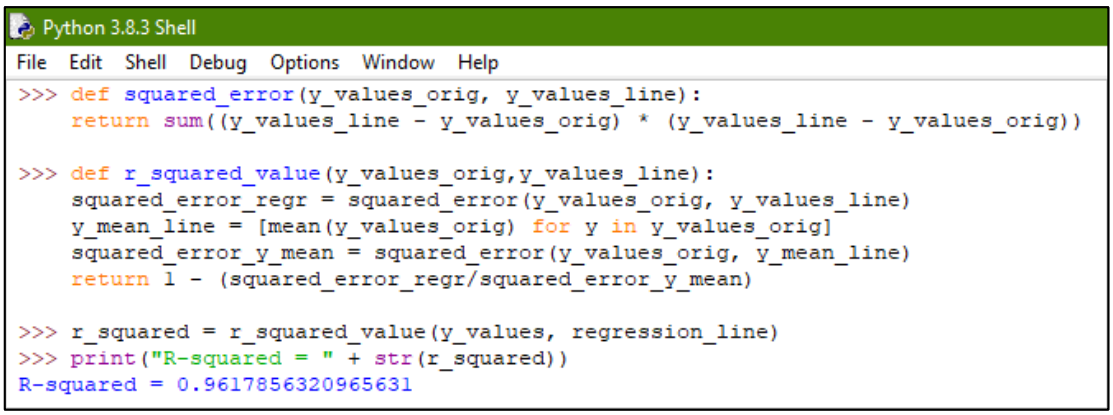

Figura 27 - Calcolo dell'R-squared su Python

Nella *figura 27* si osserva la schermata, ottenuta dalla shell interattiva di *Python*, dove si evidenzia il risultato ottenuto (R-squared = 0,96) per il caso analizzato in questa trattazione: si evince che la relazione lineare fra il fenomeno analizzato e la retta di regressione è particolarmente accurata.

A questo punto è stata calcolata l'incertezza del sensore, espressa come ***RMSE*** (”***Root-Mean-Square Error***”), che consiste in un indice di dispersione statistico, ovvero una stima della variabilità di una

popolazione di dati o di una variabile casuale intorno ad un indice di posizione come, ad esempio, la media aritmetica o una sua stima, ed ha pertanto la stessa unità di misura dei valori osservati.

Allo stesso modo, il termine "*standard deviation*" è stato introdotto da Pearson nel 1984 (assieme alla lettera greca σ), che lo rappresenta e dunque, lo *scarto quadratico medio* è la radice quadrata della varianza, la quale viene coerentemente rappresentata con il quadrato di sigma (σ^2). [20][21]

Perciò, lo *scarto quadratico medio* rivelato su una popolazione di N unità è espresso dalla relazione seguente:

$$RMSE = \sigma = \sqrt{\frac{\sum_{i=1}^{N}(\bar{y}-y_i)^2}{N}} \tag{4.13}$$

dove:

- $\bar{y}$ sono i valori previsti (appartenenti alla *regression line*);
- y_i sono i valori osservati.

Logicamente tanto più la deviazione standard è grande, tanto più i valori della distribuzione sono dispersi, viceversa, se la deviazione standard è piccola, i valori sono concentrati vicino alla media.

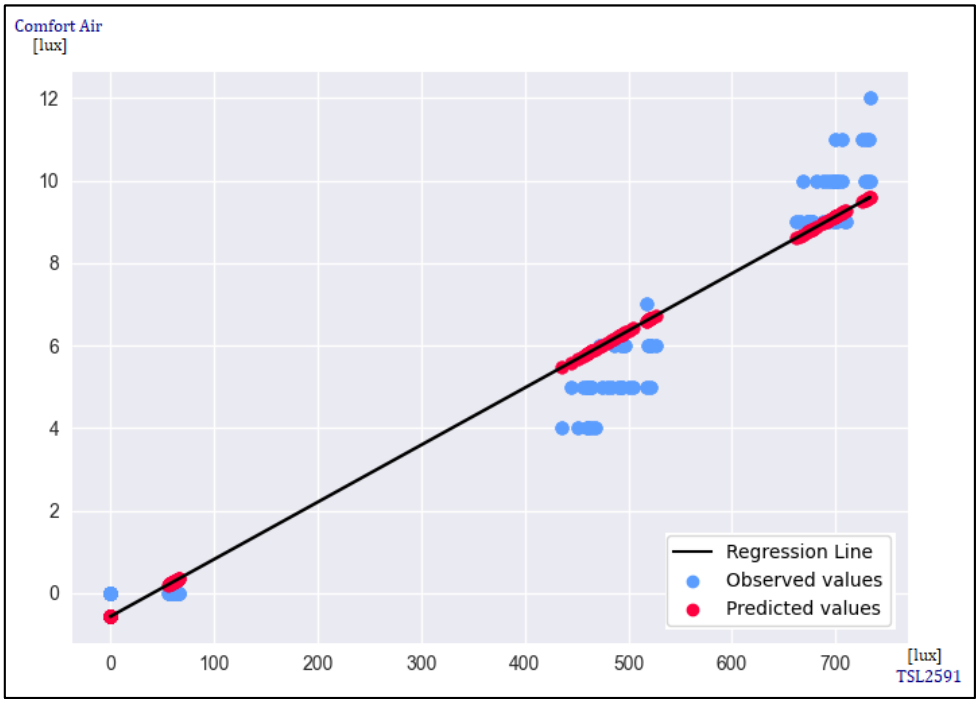

Figura 28 - Grafico con i valori osservati e previsti

Nella *figura 29*, si osserva la schermata, ottenuta dalla shell interattiva di *Python*, dove si evidenzia il risultato ottenuto (RMSE = 0,8 *lux*) per il caso analizzato in questa trattazione:

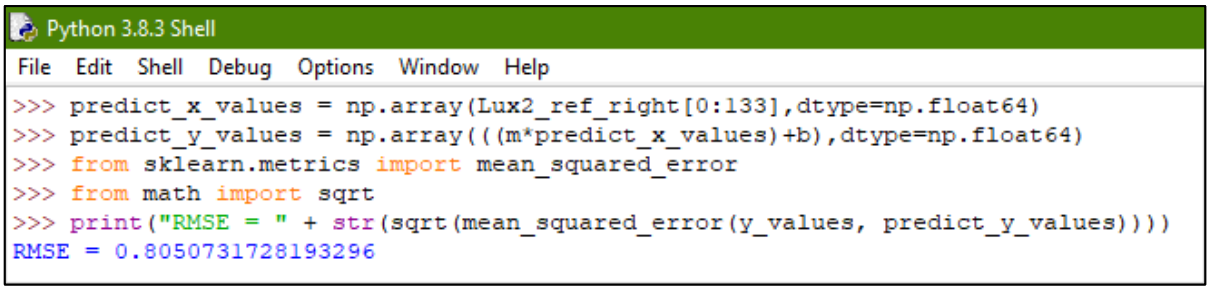

Figura 29 - Calcolo dell'RMSE su Python

Calcolato questo indice di incertezza, la calibrazione del sensore è stata completata.

4.5 Analisi e confronto con luce naturale

Completata la calibrazione del sensore di luminosità all'interno del Comfort Air, si è pensato di ripetere la medesima sperimentazione con una sola differenza: in questo caso non è stato utilizzato il faretto alogeno, con il *dimmer* connesso, per la regolazione dell'intensità luminosa, ma il *Comfort Air*, con il relativo sensore di riferimento *TSL2591*, sono stati esposti alla sola luce naturale, vicino ad una finestra, senza ulteriori influenze da sorgenti artificiali esterne.

Questo significa che i dati acquisiti, importati ed analizzati, sono stati ottenuti in assenza di sorgenti luminose artificiali.

La retta di equazione:

$$y = 0{,}12x - 4{,}86 \tag{4.14}$$

esprime la retta di regressione ottenuta evidenziando l'andamento dei valori *lux* del sensore da calibrare rispetto al sensore di riferimento, il tutto mostrato nella *figura 30*:

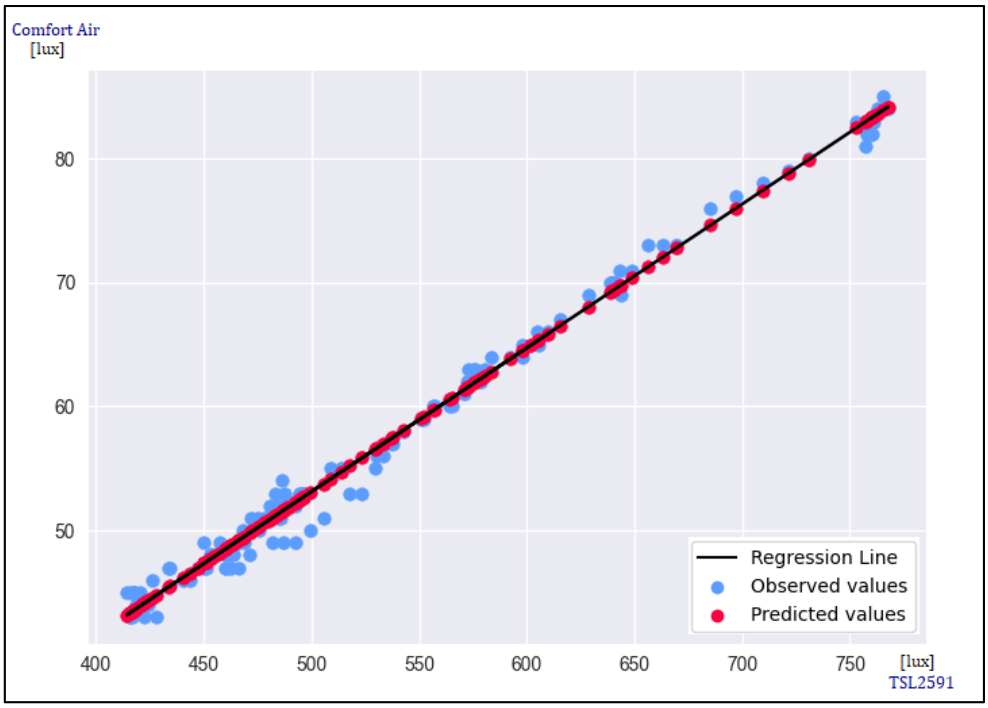

Figura 30 - Andamento dei valori lux nel caso di esposizione a luce naturale

Anche in questo caso, con l'utilizzo del software *Python*, sono stati calcolati l'*R-squared*, risultato essere pari a 0,99 e soprattutto l'indice di incertezza *RMSE*, il cui esito è pari a 1,1 *lux*.

Conclusione

Il lavoro svolto in questa tesi è stato quello di tarare e analizzare l'incertezza di un sensore di luminosità all'interno del dispositivo per la misurazione delle variabili indoor (*Comfort Air*), allo scopo di monitorare, nella maniera più accurata possibile, il comfort visivo in ambienti indoor. Il sistema *Comfort Eye* fa parte del progetto europeo *P2Endure*, ed è già stato installato all'interno in diversi luoghi come ad esempio in Polonia, più precisamente a Varsavia e a Gdynia.

Attraverso una "*web-app*", in fase di sviluppo, sarà possibile consultare i dati relativi alla variabile desiderata, come ad esempio l'intensità luminosa, la temperatura o l'umidità relativa. Come descritto in questa disamina, analizzando i due casi esposti (luce artificiale tramite faretto alogeno e luce naturale) si osserva una certa discordanza sui dati ottenuti, relativi al comfort visivo. Nel primo caso analizzato, in cui convogliava, sui dispositivi, luce artificiale, proveniente dal faretto alogeno, il sensore di luminosità installato all'interno del dispositivo *Comfort Air* ha registrato un valore di incertezza (0,8 *lux*) al di sotto del caso in cui è stata utilizzata luce naturale (1,1 *lux*).

È constatabile il fatto che fattori esterni, ad esempio l'effetto indesiderato delle sorgenti luminose artificiali in sede di sperimentazione, abbiano comportato un'interferenza dei valori, i quali sono diversi dai dati acquisiti con lo stesso sensore direttamente esposto all'intensità luminosa utilizzata nel primo caso realizzato sul banco di prova.

Pertanto, i risultati si possono ritenere soddisfacenti, seppur tenendo in considerazione che è necessario analizzare queste valutazioni a lungo termine, con misurazioni effettive, in modo che si possa constatare se il sensore risponda ugualmente in ambienti diversi e con differenti tipologie di illuminazione.

Che sia esigua o che sia eccessiva, la luce può causare un certo livello di disagio visivo, per questo è essenziale studiare e prevenire cambiamenti nei livelli di luce o nei contrasti luminosi, normalmente percepiti come abbagliamenti, che possono provocare stress e affaticamento psicofisico, nonché influisce sui ritmi circadiani (sonno e veglia). Pertanto, così come eseguito dal sistema *Comfort Eye*, è opportuno valutare, dal principio, anche questo aspetto per poi passare ad una selezione delle soluzioni di "*lighting design*" più adatte al contesto in cui ci si trova a vivere e sfruttare adeguatamente i punti d'accesso di luce solare a disposizione, allo scopo di garantire agli utenti la miglior luce naturale e/o artificiale possibile. Oltre a questo, a seconda dell'uso distinto di ogni stanza, è opportuno considerare la variazione d'uso dello spazio nei differenti momenti della giornata.

Appendice *A*

```
import asyncio
import socket
import functools
import time
import logging
import logging.handlers
import time
from datetime import datetime
import os
import re
import board
import busio
import adafruit_tsl2591

# Initialize the I2C bus.
i2c = busio.I2C(board.SCL, board.SDA)

# Initialize the sensor.
sensor = adafruit_tsl2591.TSL2591(i2c)

# You can optionally change the gain and integration time:
# sensor.gain = adafruit_tsl2591.GAIN_LOW (1x gain)
# sensor.gain = adafruit_tsl2591.GAIN_MED (25x gain, the default)
# sensor.gain = adafruit_tsl2591.GAIN_HIGH (428x gain)
# sensor.gain = adafruit_tsl2591.GAIN_MAX (9876x gain)
# sensor.integration_time = adafruit_tsl2591.INTEGRATIONTIME_100MS (100ms, default)
# sensor.integration_time = adafruit_tsl2591.INTEGRATIONTIME_200MS (200ms)
# sensor.integration_time = adafruit_tsl2591.INTEGRATIONTIME_300MS (300ms)
# sensor.integration_time = adafruit_tsl2591.INTEGRATIONTIME_400MS (400ms)
# sensor.integration_time = adafruit_tsl2591.INTEGRATIONTIME_500MS (500ms)
# sensor.integration_time = adafruit_tsl2591.INTEGRATIONTIME_600MS (600ms)


# logging.basicConfig(level=logging.DEBUG,
                    # format='%(asctime)s %(name)-12s %(levelname)-8s %(message)s',
                    # datefmt='%m-%d %H:%M',
                    # filename='/tmp/p2e_pes01cair.log',
                    # filemode='a')

# logging.basicConfig(level=logging.DEBUG)
```

```
logging.basicConfig(level=logging.DEBUG,
                    format='%(asctime)s %(name)-12s %(levelname)-8s %(message)s',
                    datefmt='%m-%d %H:%M')

#Filippo 27-06-2018
# socketHandler = logging.handlers.SocketHandler('193.205.129.218',4000)
# logging.getLogger('').addHandler(socketHandler)
hostname = socket.gethostname()
fileHandler=logging.handlers.WatchedFileHandler(filename='/tmp/'+ hostname + '_cair.log',mode='a')
logging.getLogger('').addHandler(fileHandler)

#h = logging.handlers.SysLogHandler(address = ('193.205.129.241',4101), facility=logging.handlers.SysLogHandler.LOG_USER)
#h.setFormatter(logging.Formatter('%(module)s %(message)s'))
#logging.getLogger('').addHandler(h)

def sock_accept(self, sock, fut=None, registed=False):
    fd = sock.fileno()
    if fut is None:
        fut = self.create_future()
    if registed:
        self.remove_reader(fd)
    try:
        conn, addr = sock.accept()
        conn.setblocking(False)
        logging.debug(str(conn))
    except (BlockingIOError, InterruptedError):
        self.add_reader(fd, self.sock_accept, sock, fut, True)
    except Exception as e:
        fut.set_exception(e)
    else:
        fut.set_result((conn, addr))
    return fut

def sock_recv(self, sock, n , fut=None, registed=False):
    fd = sock.fileno()
    if fut is None:
        fut = self.create_future()
    if registed:
        self.remove_reader(fd)
    try:
        data = sock.recv(n)
    except (BlockingIOError, InterruptedError):
```

```
        self.add_reader(fd, self.sock_recv, sock, n ,fut, True)
    except Exception as e:
        fut.set_exception(e)
    else:
        fut.set_result(data)
    return fut

def sock_sendall(self, sock, data, fut=None, registed=False):
    fd = sock.fileno()
    if fut is None:
        fut = self.create_future()
    if registed:
        self.remove_writer(fd)
    try:
        n = sock.send(data)
    except (BlockingIOError, InterruptedError):
        n = 0
    except Exception as e:
        fut.set_exception(e)
        return
    if n == len(data):
        fut.set_result(None)
    else:
        if n:
            data = data[n:]
        self.add_writer(fd, sock, data, fut, True)
    return fut

    async def handler(loop, conn):
    clients = set([])
    comm = set([])
    # ii=0

    while True:
        # ii+=1

        data = await loop.sock_recv(conn, 1024)

        # logging.debug("handler: ********************= %i" % (ii))

        if not data: break
        #if msg: await loop.sock_sendall(conn, msg)
        #else: break
        message=data.decode()
```

```
        if '??' in message: break

        logging.debug('MSG: {}...'.format(message))
        #logging.debug("CCCCCCCC: %s" % message[3:17])
        # logging.debug("conn: %s" % str(conn))

        while (q2.qsize()!=0):
            clients.add(await q2.get())
        for jj in clients:
            await q2.put(jj)

        if checkData(message): #Se c'e' l'admin che parla
            #logging.debug("message %s da %s" & (message[24:],message[7:23]))
            if message[2:6] == 'com*':
                await q3.put(message[6:])
                # logging.debug("Comando accodato: %s @ %s" % (message[23:],message[6:23]))
                # logging.debug("Accodato: %s" % (message[6:]))
            elif message[2:7] == 'print':
                buf=''
                for jj in clients:
                    buf=( buf + ',' + jj) if buf else jj
                # logging.debug("Scrittura: " + buf)
                if buf: await loop.sock_sendall(conn, buf.encode())

        else: #se c'è il ComfortEye che trasmette
            macaddr=message[3:17]
            # logging.debug("macaddr: %s" % str(macaddr))
            clients.add(macaddr)

            # while (q2.qsize()!=0):
                # t = await q2.get()
            # for jj in clients:
                # await q2.put(jj)

            await q1.put(message)

            # logging.debug('Writing to Queue')

    conn.close()

async def server(loop,q1,q2,q3):
    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM, 0)
    sock.setsockopt(socket.SOL_SOCKET, socket.SO_REUSEADDR, 1)
    sock.setblocking(False)
```

```
    sock.bind(('0.0.0.0', 3100))
    sock.listen(20)

    clients = set([])
    comm = set([])
    while True:
        conn, addr = await loop.sock_accept(sock)

        loop.create_task(handler(loop, conn))

def checkData(data):
    if data[0:3] == '\r\n*':
        parseCeye(data)
        return False
    elif data[0:2]=='!!':
        parseAdmin(data)
        return True

def parseAdmin(data):
    pass
    # logging.debug('Admin: '+ data)

def parseCeye(data):
    pass
    # logging.debug('Ceye: '+ data[2:])

async def consume(q1):
    while True:
        # wait for an item from the producer
        item = await q1.get()

        now = datetime.now()
        d = now.strftime("%Y%m%d")
        h = now.strftime("%H")
        m = int(now.strftime("%M"))

        if m >= 45:
            ms = "45"
        elif m >= 30:
            ms = "30"
        elif m >= 15:
            ms = "15"
        else:
            ms = "00"
```

```
        dt = d + "-" + h + ms

        if not os.path.isdir(store_dir):
           os.mkdir(store_dir)

        store_dir_d = store_dir + "/" + d
        if not os.path.isdir(store_dir_d):
           os.mkdir(store_dir_d)

        store_dir_ddt = store_dir + "/" + d + "/" + dt
        if not os.path.isdir(store_dir_ddt):
           os.mkdir(store_dir_ddt)
        regex1          =          r'((?:[0-9a-f]{2}:){5}[0-9a-f]{2})\*tem#(-?\d+),hum#(-?\d+),lux#(-
?\d+),co2#(-?\d+),voc#(-?\d+),bat#(-?\d+),'
        it = item.rstrip('\n')
        match1 = re.findall(regex1,it)
        print ("3 - ITEM dev1:", it)
        print ("3 - REGEX dev1:", regex1)
        print ("3 - MATCH dev1:", match1)
        if len(match1) != 0:
             #    MAC = str(match1[0][0])
              #  print ("4 - MAC ADDRESS dev1:", MAC)
                T = float(match1[0][1])/10
                print ("5 - T dev1:", T)
                RH = int(match1[0][2])
                print ("6 - RH dev1:", RH)
                LUX = int(match1[0][3])
                print ("7 - LUX dev1:", LUX)
                CO2 = int(match1[0][4])
                print ("8 - CO2 dev1:", CO2)
                VOC = int(match1[0][5])
                print ("9 - VOC dev1:", VOC)
                BAT = int(match1[0][6])
                print ("10 - BAT dev1:", BAT)
                stringa1  =  str(T)+"  "+str(RH)+"  "+str(LUX)+"  "+str(CO2)+"  "+str(VOC)+"
"+str(BAT)+"\r\n"
        filename=store_dir_ddt+"/p2e_"+item[3:17]+".txt"
        filename=filename.replace(":", "")
        text_file = open(filename, "a")
        text_file.write(str(int(time.time())) + stringa1)
        text_file.close()
        logging.debug("Saving: " + item + "--IN: " + filename )
        atime = 100.0;
        again = 25.0;
```

```
        TSL2591_LUX_DF    = 408.0;
        TSL2591_LUX_COEFB = 1.64;
        TSL2591_LUX_COEFC = 0.59;
        TSL2591_LUX_COEFD = 0.86;
        CH0 = 0.0;
        CH1 = 0.0;

        # Read and calculate the light level in lux.
        lux_hw = sensor.lux

        # Infrared levels range from 0-65535 (16-bit)
        infrared = sensor.infrared
        # Visible-only levels range from 0-2147483647 (32-bit)
        visible = sensor.visible
        # Full spectrum (visible + IR) also range from 0-2147483647 (32-bit)
        full_spectrum = sensor.full_spectrum
        raw_luminosity=sensor.raw_luminosity

        CH0 = raw_luminosity[0]
        print(CH0)
        CH1 = raw_luminosity[1]
        print(CH1)

        cpl = (atime * again) / TSL2591_LUX_DF
        lux1 = ( CH0 - (TSL2591_LUX_COEFB * CH1) ) / cpl
        lux2 = ( ( TSL2591_LUX_COEFC * CH0 ) - ( TSL2591_LUX_COEFD * CH1 ) ) / cpl
        lux = ( CH0 - ( 1.7 * CH1 ) ) / cpl

        filename=store_dir_ddt+"/p2e_adafruit.txt"
        filename=filename.replace(":", "")
        text_file = open(filename, "a")
        text_file.write(str(int(time.time())) + "\t"+ str(round(lux1, 2)) + "\t"+ str(round(lux2, 2))
+ "\t" + str(round(lux, 2)) + "\n")
        text_file.close()
        logging.debug("Adafruit values: LUX: " + str(lux_hw) + ", lux_c: " + str(lux)+ ",
infrared: " + str(infrared) + ", visible: " + str(visible) + ", full_spectrum: " + str(full_spectrum)
+ ".")
        logging.debug("Adafruit values: LUx1: " + str(lux1) + ", lux_2: " + str(lux2)+ ", infrared:
" + str(infrared) + ", visible: " + str(visible) + ", full_spectrum: " + str(full_spectrum) + ".")
        logging.debug("Saving: " + str(lux) + "--IN: " + filename )

        # if item is None:
            # logging.debug('None items') # the producer emits None to indicate that it is done
            # break
```

```
        # process the item
        # logging.debug('consuming item {}...'.format(item))
        # simulate i/o operation using sleep
        #await asyncio.sleep(0.1)
        # logging.debug("La coda q1 in producer ha %i elementi." % (q1.qsize()))

EventLoop = asyncio.SelectorEventLoop
EventLoop.sock_accept = sock_accept
EventLoop.sock_recv = sock_recv
EventLoop.sock_sendall = sock_sendall
loop = EventLoop()
asyncio.set_event_loop(loop)

q1 = asyncio.Queue(loop=loop) #Coda temperature
q2 = asyncio.Queue(loop=loop) #Coda clients
q3 = asyncio.Queue(loop=loop) #Coda comandi

store_dir = "/tmp/pesto"

try:
    #loop.run_until_complete(server(loop,q1,q2,q3))
    producer_coro = server(loop,q1,q2,q3)
    consumer_coro = consume(q1)
    loop.run_until_complete(asyncio.gather(consumer_coro,producer_coro))

except KeyboardInterrupt:
    pass
finally:
    loop.close()
```

Appendice *B*

```
>>> import re
>>> regex1 = r"(-?\d+.?\d+)\s(-?\d+)\s(-?\d+)\s(-?\d+)\s(-?\d+)\s(-?\d+)"
>>> filedata1 = (r"C:\Users\Federico\Desktop\raw_data1.txt")
>>> lines = [line.rstrip('\n') for line in open(filedata1)]
>>> t = ([])
>>> rh = ([])
>>> lux = ([])
>>> co2 = ([])
>>> voc = ([])
>>> bat = ([])
>>> for riga in lines:
        print ("riga: " + riga)
        match = re.findall(regex1, riga)
        print ("match: " + str(match))
        if len(match)!=0:
                t.append(float(match[0][0]))
                rh.append(float(match[0][1]))
                lux.append(float(match[0][2]))
                co2.append(float(match[0][3]))
                voc.append(float(match[0][4]))
                bat.append(float(match[0][5]))

>>> regex2 = r"(-?\d+.?\d+)\s(-?\d+.?\d+)\s(-?\d+.?\d+)"
>>> filedata2 = (r"C:\Users\Federico\Desktop\raw_data2.txt")
>>> lines = [line.rstrip('\n') for line in open(filedata2)]
>>> Lux1_ref_wrong = ([])
```

```
>>> Lux2_ref_wrong = ([])
>>> Lux_ref_wrong = ([])
>>> for riga in lines:
        print ("riga: " + riga)
        match = re.findall(regex2, riga)
        print ("match: " + str(match))
        if len(match)!=0:
                Lux1_ref_wrong.append(float(match[0][0]))
                Lux2_ref_wrong.append(float(match[0][1]))
                Lux_ref_wrong.append(float(match[0][2]))


>>> Lux1_ref = [0 if i < 0 else i for i in Lux1_ref_wrong]
>>> Lux2_ref = [0 if i < 0 else i for i in Lux2_ref_wrong]
>>> Lux_ref = [0 if i < 0 else i for i in Lux_ref_wrong]
>>> lux_right = [item for index, item in enumerate(Lux_ref) if index %6 !=0]
>>> Lux1_ref_right = [item for index, item in enumerate(Lux1_ref) if index %6 !=0]
>>> Lux2_ref_right = [item for index, item in enumerate(Lux2_ref) if index %6 !=0]
>>> print(len(lux_right))
>>> print(len(Lux2_ref_right))


>>> from statistics import mean
>>> import numpy as np
>>> x_values = np.array(Lux2_ref_right[0:133],dtype=np.float64)
>>> y_values = np.array(lux_right[0:133],dtype=np.float64)
>>> def best_fit_line(x_values,y_values):
        m = (((mean(x_values)*mean(y_values)) - mean(x_values*y_values)) /
((mean(x_values)*mean(x_values)) - mean(x_values*x_values)))
        b = mean(y_values) - m*mean(x_values)
```

```
        return m, b
>>> m, b = best_fit_line(x_values, y_values)
>>> print("regression line: " + "y = " + str(round(m,2)) + "x + " + str(round(b,2)))
>>> regression_line = [(m*x)+b for x in x_values]

>>> def squared_error(y_values_orig, y_values_line):
        return sum((y_values_line – y_values_orig) * (y_values_line – y_values_orig))

>>> def r_squared_value(y_values_orig,y_values_line):
        squared_error_regr = squared_error(y_values_orig, y_values_line)
        y_mean_line = [mean(y_values_orig) for y in y_values_orig]
        squared_error_y_mean = squared_error(y_values_orig, y_mean_line)
        return 1 - (squared_error_regr/squared_error_y_mean)

>>> r_squared = r_squared_value(y_values, regression_line)
>>> print("R-squared = " + str(r_squared))

>>> x = [(regression_line - b)/m for regression_line in y_values]
>>> X = np.array(x, dtype=np.float64)
>>> Y = np.array(Lux2_ref_right[0:133])
>>> errore_sistematico =np.mean(X - Y)
>>> print("Errore sistematico = " + str(errore_sistematico))

>>> from sklearn.metrics import mean_squared_error
>>> from math import sqrt
>>> predict_x_values = np.array(Lux2_ref_right[0:133],dtype=np.float64)
>>> predict_y_values = np.array(((m*predict_x_values)+b),dtype=np.float64)
```

```
>>> print("RMSE = " + str(sqrt(mean_squared_error(y_values, predict_y_values)))


>>> plt.scatter(x_values, y_values,color='#5b9dff',label='Observed values')

>>> plt.scatter(predict_x_values, predict_y_values, color='#fc003f', label="Predicted values")

>>> plt.plot(x_values, regression_line, color='000000', label='Regression Line')

>>> plt.legend(loc=4)

>>> plt.show()
```

Bibliografia e sitografia

[1] https://www.benessereambientale.com/benessere-ambientale-indoor.html

[2] A cura di: Dr. Marco Arnesano, Prof. Gian Marco Revel, *P2Endure, un nuovo progetto per lanciare soluzioni innovative di deep renovation degli edifici*, UNIVPM, 2016.

[3] A cura di: Prof. Gian Marco Revel, Dr. Marco Arnesano, Ing. Filippo Pietroni, Ing. Lorenzo Zampetti, *Comfort Eye: il benessere e l'efficienza*, UNIVPM, 2016.

[4] A cura di: Prof. Filippo De Rossi, *Richiami di benessere termo-igrometrico e qualità dell'aria*, UNINA, 2014.

[5] https://www.robotstore.it/Raspberry-Pi-Zero-Versione-1-3

[6] Vinces, *Comfort visivo e illuminazione interna*, Biblus-Net, 2016.

[7] UNI, Norma Italiana UNI 12464-1:2011, *Luce e illuminazione - Illuminazione dei posti di lavoro*, 2011.

[8] *Principi di illuminotecnica, studio e progettazione tecnica della luce migliore*. Giannetto Venturi, TuttoLuce, 2014.

[9] *Lezioni di illuminotecnica. Pietro Palladino*, Tecniche Nuove, 2002.

[10] *Sensore di luce ambiente TSL2591*. Digi-key Electronics, 2019.

[11] Marco Lai, *Come usare il sensore di luminosità TSL2591*, Logicaprogrammabile.it, 2015.

[12] Stefano Selleri, Luca Vincetti, Annamaria Cucinotta, *Componenti ottici e fotonici*, Esculapio, 2012.

[13] Massimo Pappalardo, *Elettronica. Fondamenti dei dispositivi e dei circuiti*, Franco Angeli, 2004.

[14] A cura di Luigi Nasone, *La giunzione p-n*, Chimitutor, 2017.

[15] A cura di Know How, *Arduino vs Raspberry Pi: le differenze*, Digital Guide Ionos, 2019.

[16] Tony DiCola, *Circuit Python module for the TSL2591 precision light sensor,* Tony DiCola for Adafruit Industries, 2017.

[17] Ernest O. Doebelin, *Strumenti e metodi di misura (Seconda edizione)*, McGraw-Hill Education, 2008.

[18] H. D. Young, *Statistical Treatment of Experimental Data*, McGraw-Hill, 1962.

[19] James Stock, Mark Watson, *Introduzione all'econometria*, Milano, Pearson Education, 2005

[20] Karl Pearson, *On the dissection of asymmetrical frequency curves*, 1894

[21] UNI, Norma italiana UNI 4723:1984, *Metodi statistici per il controllo della qualità. Termini, simboli e definizioni*, 1984