Politecnico di Torino

Collegio di Ingegneria Gestionale – Classe L/8
Corso di Laurea in Ingegneria Gestionale

Implementazione software per il disegno in AutoCad d'impianti di sollevamento

Relatore:
Fabrizio Lamberti

Candidato:
Fabiano Vaglio

Anno accademico 2022-2023

INDICE

1. Analisi dei componenti di un impianto di sollevamento

Nel dettaglio di questa tesi andremo ad analizzare le componenti di un impianto di sollevamento attraverso alcune piantine del vano, in quanto queste ultime sono state l'oggetto di studio.
Nel seguente capitolo si andrà ad esaminare l'organizzazione dello spazio all'interno di un vano, differenziando le componenti ed indicandone le funzioni pratiche.

1.1 Componenti da pianta

In questa prima parte andremo a capire, con riferimento alla *Figura 1*, l'organizzazione generale degli spazi all'interno del vano destinato all'impianto di sollevamento.
Le componenti principali di una sezione orizzontale del vano (chiamata pianta) che andremo a trattare sono: Vano, Cabina, Contrappeso, Arcata, Soglia e Parti elettriche quali pulsantiera di piano e di cabina

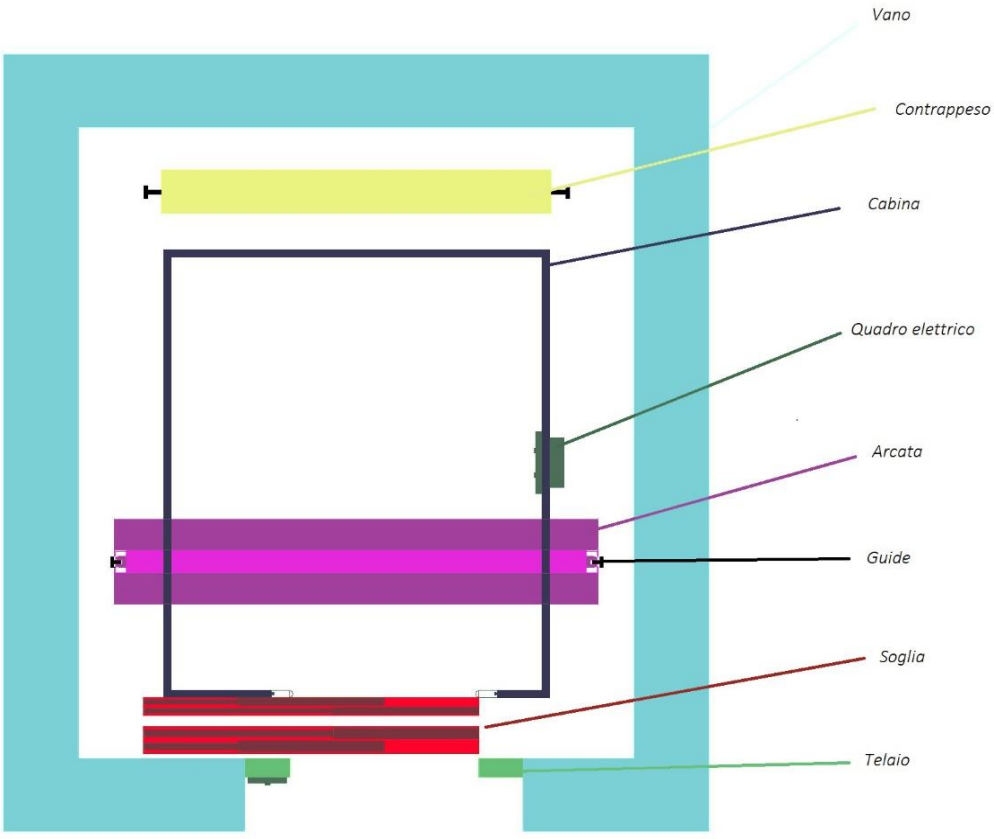

Figura 1. Pianta di un generico impianto con, in evidenza, relative parti

1.2 Vano

Il vano è il volume delimitato dal fondo fossa, corsa e testata, nella quale si sviluppa per intero, esclusivamente l'impianto di sollevamento. Nella *Figura 2*, si può osservare un esempio di una sezione verticale di un impianto. Il vano, solitamente in calcestruzzo, deve rispettare requisiti di sicurezza strutturale e deve essere conforme ai regolamenti nazionali e regionali che hanno come fine ultimo, oltre che garantirne la sicurezza, quello di abbattere le barriere architettoniche.

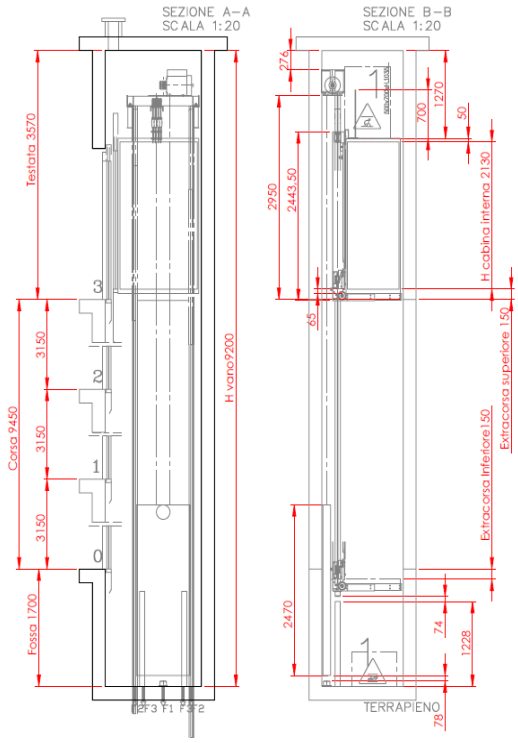

Figura 2. Sezione verticale di un vano

Quindi, in particolare, il vano deve possedere caratteristiche di robustezza e solidità atte a resistere ai carichi di progetto, oltre che ai sismi o, nel caso di impianti posti all'esterno, di vento o neve.

Il vano comprende, quindi, anche la testata, che parte dal pavimento dell'ultimo piano fino alla fine del vano, e la fossa, la quale parte dalla soglia del piano più inferiore fino alla fine del vano corsa stesso. In quest'ultima possono essere presenti altri componenti, come gli ammortizzatori, i quali hanno lo scopo di dissipare l'energia cinetica che la cabina accumula in un'eventuale malaugurata caduta.

1.3 Cabina

La cabina rappresenta la parte decorativa e visibile all'utente, è pertanto fondamentale considerare che in essa, oltre che allo scopo pratico fondamentale, vince l'esigenza di curarne il design nel dettaglio. Tuttavia, non bisogna dimenticare fattori come l'accessibilità e la dimensione, che permettono a tutti i passeggeri di usufruirne in maniera equa e semplice.

1.4 Contrappeso

Esistono due tipi di contrappeso, e si distinguono in “contrappeso a guide” e “contrappeso a bordiglioni”. Quest’ultimo, a differenza di quello a guide, non è molto usato al giorno d’oggi, poiché è caratterizzato dal fatto che, a guidarlo per la sua corsa ci sono delle funi e non pattini e guide. Di diverse dimensioni e materiali, il contrappeso, ha la funzione primaria di bilanciare il peso della cabina, assicurando la trazione tra la puleggia di trazione e le funi di sospensione. Inoltre, riduce il lavoro per sollevare il carico da parte dell’argano dato che, quando la cabina sale, lui scende. Il rapporto di peso tra cabina e contrappeso non è sempre 1:1, anzi, è quasi sempre sbilanciato. Per determinare il rapporto di peso tra contrappeso e cabina, si lavora statisticamente in base all’utilizzo che l’impianto andrà a soddisfare. Ad esempio, in un condominio, la cabina viaggerà per la maggior parte delle volte vuota (circa ½ delle volte), mentre negli ospedali l’uso dell’ascensore sarà, la maggior parte delle volte, con carico. Questo vuol dire che la massa del contrappeso, nel caso del condominio, dovrà essere più vicina alla massa della cabina vuota e, nel caso dell’ospedale, più vicina alla massa della cabina a pieno carico. Questo comporta inevitabilmente che, entro certi limiti, il contrappeso pesi di più della cabina vuota, ma non con un rapporto di peso fisso per tutti gli impianti.

1.5 Impianti elettrici e oleodinamici

I due tipi di impianti più diffusi sono l’elettrico e l’oleodinamico.
L’impianto oleodinamico o idraulico (*Figura 3*), è utilizzato in caso di impianti con poche fermate, poco traffico e senza necessità di velocità elevate. Esso è caratterizzato da uno o più pistoni idraulici telescopici posizionati in centro o in taglia all’arcata. Data l’economicità per l’istallazione e la versatilità, dovute al fatto che l’impianto non necessita di un locale macchine o di un contrappeso, lo rende ottimo da adattare a edifici da ristrutturare o esistenti, soprattutto se gli spazi nel vano sono limitati. D’altro canto, però, è oneroso nei costi d’esercizio, siccome presenta elevati costi elettrici per il funzionamento e parecchi costi dovuti alla manutenzione. Per quest’ultima è infatti necessario, ogni 10 anni, cambiare i tubi flessibili e l’olio, i quali sono difficili da smaltire per via delle componenti inquinanti.

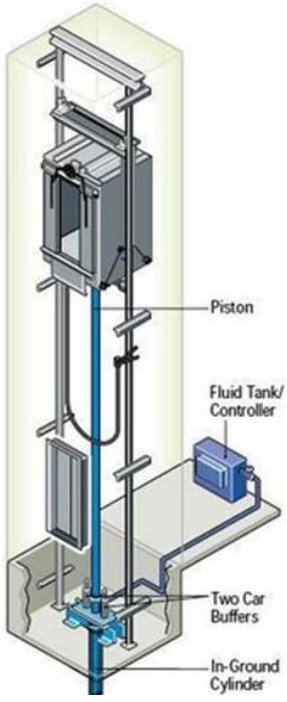

Figura 3. Impianto Oleodinamico

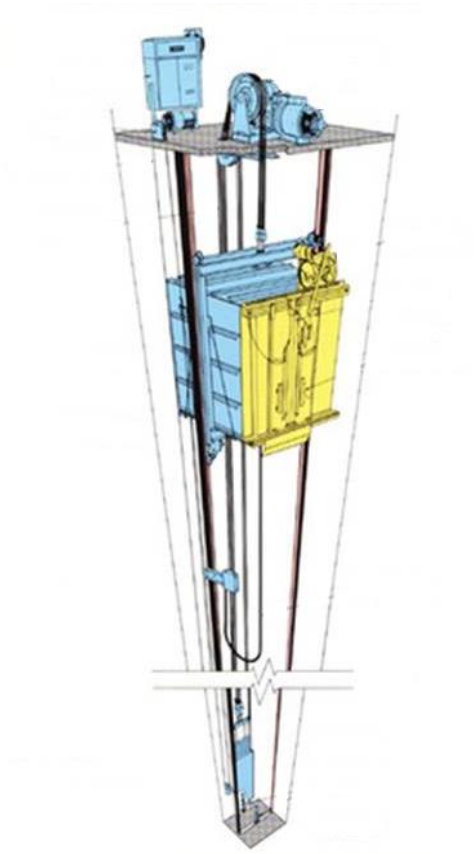

Figura 4. Impianto elettrico

Il movimento di un impianto elettrico (*Figura 4*) è dipendente da un argano, solitamente posizionato nel locale macchina, ad uso esclusivo, sopra la testata del vano. Negli impianti elettrici è sempre presente il contrappeso, che bilancia il peso della cabina fornendo sufficiente attrito sulla puleggia dell'argano. La puleggia non è altro che una ruota con numerose scanalature a "V" o a "C" all'interno delle quali scorrono le funi; essa, girando in senso orario o antiorario, permette la salita o la discesa della cabina. A differenza degli impianti oleodinamici, possono raggiungere elevate velocità e non hanno limiti riguardanti il numero di fermate, la portata o il traffico. Inoltre non necessitano dello smaltimento dei materiali inquinanti durante la manutenzione, rendendoli a tutti gli effetti più ecologici, ma costosi al momento dell'installazione.

1.6 Arcata

L'arcata è la struttura metallica entro cui è collocata la cabina. Essa scorre, con l'ausilio di pattini, su apposite guide metalliche montate verticalmente lungo tutto il vano, con lo scopo di guidare e dirigere la corsa dell'arcata e, di conseguenza, della cabina.

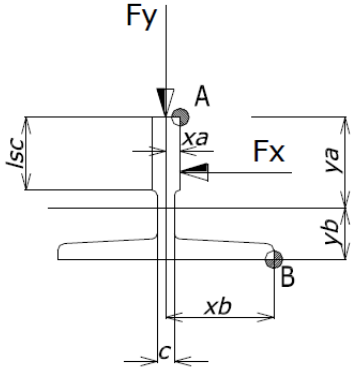

Figura 5. Guida T82/B

Il profilo della sezione delle guide, lavorate a macchina nella trafilatura a freddo o negli stampi, è generalmente a "T", come si vede in *Figura 5*. Invece, i pattini, hanno la forma complementare a "C" e, solitamente, hanno la guarnizione interna in materiali plastici, come il Perlon, e l'esterno in diversi materiali, come ghisa, acciaio e alluminio, che variano a seconda della portata e dell'uso dell'impianto. L'arcata può essere di due tipi: ad "Anello" o a "Sedia", a seconda della lunghezza della corsa e dallo spazio disponibile nel vano.

L’arcata ad Anello è detta così perché montata tutto attorno alla cabina e necessita di guide su pareti opposte, mentre l’arcata a Sedia è caratterizzata da una forma che comprende un’intelaiatura di sostegno sulle guide da un lato, e una struttura posta sotto la cabina per reggerla. L’arcata a sedia è vantaggiosa quando il vano non è interamente composto da pareti portanti, ma da un lato è, ad esempio, in cemento armato e i restanti lati sono inesistenti o da vetro, perché permette di montare le guide sulla parete in grado di reggerle e non su più pareti come con l’arcata ad anello.

1.7 Soglia

Con il termine soglia si intende l’unione di tre spazi: quello dove scorrono le porte di piano, lo spazio in cui scorrono le porte di cabina e lo spazio di distanza fra le due.

Si possono però distinguere 3 tipi di soglia:

Soglia automatica, come si osserva dalla piantina di *Figura 6*, dove sia sul piano sia sulla cabina sono presenti porte di tipo automatizzato, che quindi non richiedono l’intervento manuale di un qualsiasi utilizzatore. Questo tipo di soglia è la migliore, se consideriamo l’utilizzo degli spazi, perché permette di utilizzare meglio l’area interna alla cabina rendendo così più agevole l’utilizzo ad una persona in sedia a rotelle. Possiamo poi distinguerle ulteriormente in base alla direzione nella quale le porte si chiudono. Spesso il vano costringe la soglia a chiudersi in una direzione (sinistra o destra) per questioni di spazio oppure, come in questo caso, l’apertura è centrale e le porte si aprono 2 per lato.

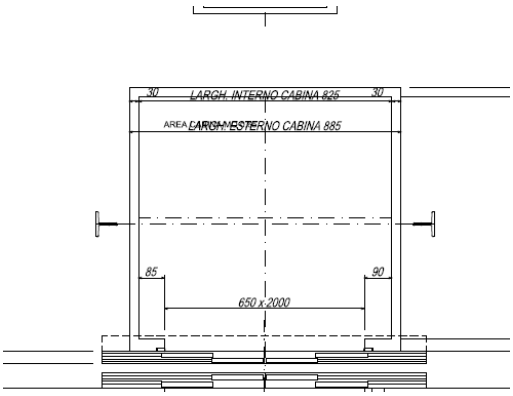

Figura 6. Soglia Automatica

La soglia semi-automatica (*Figura 7*) è caratterizzata dall’uso di una porta automatica sulla cabina, che come nel caso precedente, non compromette l’uso degli spazi interni, e di una porta manuale sul piano che però, oltre a non essere pratica per le persone disabili, diminuisce la luce (misura dello spazio di apertura) a causa del telaio sulla porta. Questa soluzione è adottata

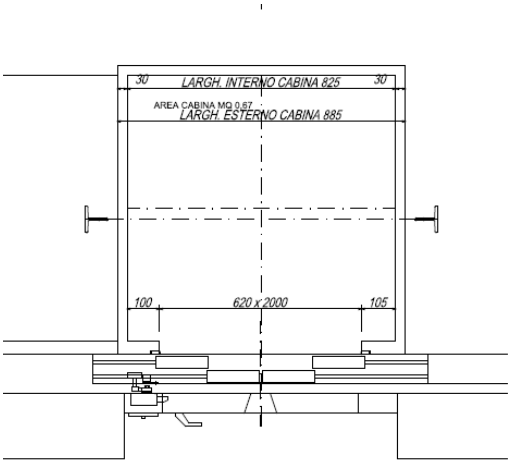

Figura 7. Soglia semi-automatica

sull'ammodernamento di impianti vecchi dove, invece, le soglie erano a battenti, così da risparmiare il costo delle porte di piano cambiando solo quelle interne.

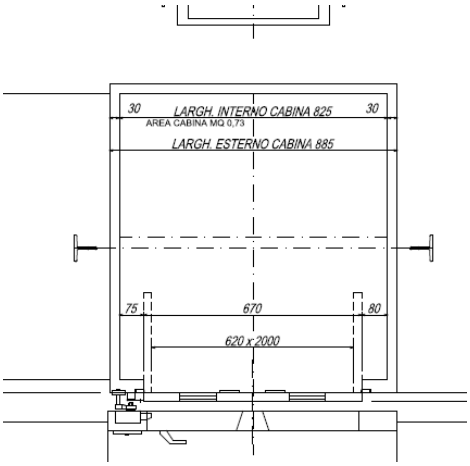

Figura 8. Soglia a battenti

La soglia a battenti (*Figura 8*) si vede spesso negli impianti più vecchi ed è caratterizzata da due porte ad apertura manuale sulla cabina e di una porta, sempre manuale, sul piano. Questa soluzione è la peggiore in termini di utilizzo degli spazi, poiché sacrifica sia l'area interna della cabina, perché lo spazio utilizzabile non comprende l'area di apertura delle porte, sia la luce di apertura per via dello spazio laterale occupato dal telaio. Durante i lavori di ammodernamento, è infatti sempre proposta una soglia completamente automatica. Lo spazio che intercorre tra le soglie è sempre fondamentale considerarlo, siccome la distanza tra le pareti del vano e la soglia della cabina non è sempre la stessa. Naturalmente, questo problema è più evidente in edifici molto vecchi o molto alti, dove la possibilità di errore dovuta ai mezzi di costruzione o all'altezza è più alta. Le pareti del vano sono irregolari, quindi, tra un piano ed un altro, potrebbero esserci diverse dimensioni di profondità, tuttavia l'impianto deve sempre essere perfettamente verticale. Questo problema, seppur minimo, in quanto si tratta di irregolarità nell'ordine di 0-80mm, comporta che in ogni piano si abbia un riporto di soglia, ovvero una parte di soglia di piano sporgente all'interno del vano. Lo spazio che intercorre tra le soglie permette di semplificare la questione assottigliando il riporto sul piano o addirittura, nel caso di differenze minime, di eliminarlo.

1.8 Quadro e componenti elettrici

Con il termine quadro elettrico si va ad indicare il componente presente nel locale macchina, il quale consente all'elevatore di trasportare le persone in tutta sicurezza e comfort. I moderni quadri hanno al loro interno un inverter, che riduce l'assorbimento di corrente elettrica e sollecita meno i pezzi meccanici interessati. Attraverso questo strumento, l'impulso che si riceve dalla pulsantiera dell'ascensore e dalle richieste dai pulsanti di piano, coordina le fermate e decide quale azione svolgere e in che ordine, al fine di garantire la corsa nella maniera più efficiente.

2. Strumento AutoCad e file .dxf

In questo capitolo andremo a vedere quello che è l'argomento cardine sul quale è sviluppata la tesi. L'analisi dei file di costruzione .dxf e lo studio di come sono effettivamente scritti e strutturati.

2.1 Scelta dall'analisi dei file AutoCad e la divisione del file .dxf

Ogni creazione generata attraverso il Software di AutoCad ha la possibilità di essere salvato in 2 formati: DWG e DXF. Il primo è un file binario rende impossibile l'interpretazione del codice in ogni maniera, mentre i file salvati nel secondo formato sono interpretabili, in quanto, per rendere migliore lo scambio di disegni tra diversi software, il file .dxf è scritto in codice ASCII, quindi leggibile con un qualsiasi editor di testo, ed è proprio in questo modo che analizzeremo i file.
Attraverso il file messo a disposizione nell'archivio di Autodesk "autocad_2012_pdf_dxf-reference_enu.pdf "[1] mi è stato possibile analizzare i criteri con i quali un file .dxf viene creato.
Per capire la lettura del file di disegno e comprendere la funzione delle diverse parti, andremo ad analizzarle singolarmente. Ogni file è suddiviso in 7 parti, dette sezioni:

- Header
- Classes
- Tables
- Blocks
- Entities
- Objects
- Thumbnailimage

Ogni sezione non è essenziale, ma è importante capirne il ruolo per poi, nel caso, escluderle o non modificarle. E' invece importante l'ordine in cui vengono chiamate, ovvero quello con cui sono dichiarate sopra.

2.2 Header

La parte di Header contiene le informazioni generali del disegno. Consiste in un database AutoCad in versione numerica e in un numero di variabili di sistema. Ogni parametro contiene il nome della variabile e il suo associato valore. All'interno dei file AutoCad che andremo a

generare sono presenti 133 variabili, con il fine di rendere possibile la creazione del disegno e per definire valori di default o verificarne l'ammissibilità. Per capire meglio la struttura, andremo a vedere le più importanti. Ecco alcuni esempi di come è strutturata la parte di header:

```
“0
SECTION
2
HEADER
 9
$ACADVER
 1
AC1009
 9“
```

Questa è la primissima parte di ogni file, e fa capire come ogni indicazione sia divisa su ogni riga, così da non permettere confusioni tra numeri diversi e numeri a più cifre.
“0” è all'inizio di ogni file e alla fine di ogni sezione, come ad indicare lo stacco tra una sezione ed un'altra. “SECTION” e “2” stanno ad indicare l'inizio di una nuova sezione, e si ripeteranno anche nell'introduzione di tutte le altre.
Con la dicitura “HEADER”, si indica l'inizio della sezione e se ne specifica il nome; per l'appunto, “header”.
Il “9”, indica il fatto che il file conterrà gruppi di valore “String” il quale limite è di 2049 caratteri per byte, alla fine della linea comprendente il 9. Questo significa che, in seguito ad ogni 9, si avrà nella linea successiva una stringa indicante una determinata funzionalità.
Con “$ACADVER”, si indica che i caratteri in seguito indicheranno la versione di disegno scelta, di database di disegno AutoCad .
“1” è l'identificatore della variabile prima citata, mentre ”AC1009” è il valore ed indica una versione tra le più vecchie (1992) di file .dxf. La scelta di questo formato è dovuta al fatto che, oltre alla sua semplicità, è supportato da tutte le versioni AutoCad (sia windows che mac), anche da quelle gratuite che tutti hanno a disposizione.
Il “9” finale introduce il fatto che nella riga successiva apparirà un'altra stringa e così via per ogni variabile che s'intende introdurre per le proprie esigenze.
La funzione “$ANGDIR”, seguita dal numero identificativo “70” e l'indicazione “0”, indica che gli angoli andranno considerati in direzione antioraria, mentre se avessimo indicato “1”, sarebbero stati considerati in senso orario ma, siccome in trigonometria e su AutoCad si considerano di default in senso anti orario, ho lasciato “0”.

La funzione “$TEXTSIZE”, preceduta dal “9” che la introduce, indica l’altezza dei caratteri di testo, se introdotti di default. In generale, l’introduzione di default non comporta che durante la vera e propria definizione delle dimensioni non sia possibile introdurne un’altra che si preferisce, ma semplicemente che, in caso di mancato inserimento, ne verrà considerata una definita. Questo è utile nel caso in cui ci si dimentichi di specificare la dimensione, così che all’interno del file il messaggio non si perda, ma che piuttosto sia solo mal dimensionato.
La funzione “$TEXTSTYLE” seguita da “7” e “STANDAR”, indica lo stile di scrittura all’interno del file, ovvero quello standard predefinito da Autocad.
Una funzione indispensabile all’interno dei file per la creazione di disegni delle piantine è “$CECOLOR” seguita dal numero identificativo “62” e dall’indicazione “256”, la quale permette di introdurre 256 colori all’interno dei nostri file. Questo è importante siccome si è deciso che all’interno di questi file, per evitare confusione nei disegni, tutte le linee di disegno sarebbero state nere come da default e le quote invece rosse.
Ogni sezione termina con la dicitura:

“ENDSEC
  0”

La quale sta ad indicare la fine della sezione, mentre lo 0 serve a separare una sezione da un’altra.

2.3 Classes

La parte classes contiene tutte le informazioni per le classi definite dall’applicazione, le cui istanze compaiono nei blocchi, entità e oggetti della banca dati. Una definizione di classe è fissata in modo permanente nella gerarchia delle classi.
Dato che per la creazione dei disegni non c’è stato bisogno di usare la sezione Classes, nei file che andremo a generare è stata omessa.

2.4 Tables

Contiene le definizioni per le seguenti tabelle di simboli: APPID, ovvero la tabella d’identificazione dell’applicazione, BLOCK_RECORD, cioè la tabella di riferimento del blocco, DIMSTYLE la tabella di dimensione dello stile, LAYER la tabella dei livelli, LTYPE la tabella

dei tipi di linee, STYLE la tabella degli stili di testo, UCS tabella del sistema di coordinate utente, VIEW per vedere la tabella e VPORT la tabella di configurazione della vista.
Ogni tabella verrà definita, anche se non nell'ordine qui riportato.
La sezione inizia quindi con:

"SECTION
  2
TABLES
  0
TABLE
  2
APPID"

La dicitura "SECTION" "2" indica l'inizio di una nuova sezione, "TABLES" indica il tipo di sezione che sta per iniziare, "0", "TABLE" e successivamente "2" per indicare l'inizio della tabella, "APPID" per il nome della tabella che si inizia.
Nel caso dei file che abbiamo bisogno di creare, nessuna delle tabelle è stata modificata dallo standard di AutoCad,, ad esclusione della tabella LTYPE.
Nella tabella LTYPE sono stati aggiunti 3 tipi di linea: la linea continua normale, la linea "tratteggiata", utile ad indicare all'interno della pianta oggetti nascosti da altri e la linea "tratto punto" per indicare all'interno del disegno quelle che sono linee di mezzeria, interasse, o linee di costruzione dei disegni. Il codice che mi ha permesso di inserire una delle tre linee è:

"LTYPE
  2
CONTINUOUS
  70
    0
  3
Solid line
  72
  65
  73
    0
  40
0.0
  0"

Con questo si indica tutto ciò che è necessario scrivere per inserire, nella tendina “Proprietà” in alto nell’interfaccia del programma AutoCad, la linea Continuous. Di conseguenza, ciò permette di inserirla quando si andrà a definire le linee nella sezione “Entities” che tratteremo in seguito.
Il numero “70” è il numero di flag con cui iniziare queste diciture, “3” è il numero descrittivo per il tipo di linea mentre “Solid line” è la stringa descrittiva. Il “72” è il codice di allineamento il cui valore è “65” che, in codice ASCII è “A”. Il “73” indica il numero di elementi di linetype, “40” la lunghezza totale del modello e l’ultimo “0” chiude la dicitura di questa linea.

“LTYPE
2
ACAD_ISO04W100
70
0
3
Tratto lungo punto ISO ____ . ____ . ____ . ___
72
65
73
4
40
30.0
49
24.0
49
-3.0
49
0.0
49
-3.0
0”

Con questo codice, invece, si definisce la linea “tratto punto” che nel database delle linee di AutoCad è chiamata “ACAD_ISO04W100”. A differenza della linea continua, in questo caso non si può inserire “Solid line” perché essa non è continua e, per tanto, si inserisce l’andatura della linea “Tratto lungo punto ISO ____ . ____ . ____ . ___” . Dopo il “40” che, come prima, indica la lunghezza del modello, ci sono una serie di “49” con i relativi valori che indicano la lunghezza del trattino, punto o spazio (una voce per elemento). Per la linea tratteggiata il

codice è il medesimo, ma si differenzia per il fatto che la linea si chiamerà "ACAD_ISO02W100", e ci sarà scritto "Tratto ISO __ __ __ __ __ __ __ __ __ __ __ __". Infine, non avendo i punti da definire ma solo trattini e spazi, avrà due valori di "49" in meno.

2.5 Block_record

Contiene la definizione del blocco e le entità di disegno che creano ogni riferimento di blocco nel disegno. Questa sezione non è stata modificata, ma lasciata com'era per intero di default.

2.6 Entities

Questa sezione contiene gli oggetti grafici (entità) nel disegno, compresi i riferimenti di blocco.
In questa sezione si avrà il vero e proprio sviluppo di tutto quello che è il progetto. Infatti, è proprio qui che le entità inserite definiscono ciò che è o meno il disegno, distinguendole per vari fattori. E' quindi importante far attenzione ai valori che si inseriscono perché, oltre a quelli che definiscono l'entità, ci sono quelli che vanno a definirne le caratteristiche.
Ricordiamo che i valori interi introdotti in questa sezione, sono in millimetri, dato che nella sezione di header l'abbiamo definito come default value.
Come per ogni altra sezione, dopo la conclusione data dal "ENDSEC" e "0" della sezione precedente, si hanno le 4 righe a definire l'inizio della nuova sezione:

"SECTION
2
ENTITIES
0"

"2" e "ENTITIES" sono utilizzate per introdurre il nome della nuova sezione: Entities, mentre lo "0" è inserito per separare e, dopo di che, si introducono le entità specifiche una di seguito all'altra. A specificare la divisione tra un'entità ed un'altra c'è sempre uno "0".
Per le piantine del vano che andremo a trattare, avremo bisogno solo dei seguenti oggetti grafici: Line, Solid, Arc, Circle e Text anche se AutoCad prevede il disegno di molte più entità definite in diversi modi.

Per l’entità “LINE” il codice è:

“LINE
 5
1
 8
1
 6
CONTINUOUS
 10
0
 20
10
 30
0
 11
10
 21
10
 31
0
 0”

La stringa “LINE” introduce il fatto che l’entità introdotta è una linea, il numero “5” è il numero di handle fisso che non è omissibile, l’“8” è identificativo del layer, mentre i due numeri “1” sono identificativi dell’entità. Ciò comporta che questa sia la prima e unica, tutte le seguenti non potranno avere il numero 1 nelle stesse posizioni, altrimenti AutoCad si blocca. Il numero “6” introduce il tipo di linea che si vuole definire nella riga seguente, in questo caso “CONTINUOUS”, ma avrebbe potuto avere scritto “ACAD_ISO04W100” per la linea tratto punto oppure “ACAD_ISO02W100” per la linea tratteggiata. Nel caso in cui la linea abbia un colore da definire, si devono aggiungere in seguito al tipo di linea, un “62” che introduce il fatto che nella prossima riga sarà indicato il numero del colore e, per esempio, “1” per il rosso e “4” per il ciano (abbiamo tutti e 256 i colori definiti). Se il codice indicante il colore viene trascurato, il software, dato che nella sezione di header è stato definito il default, introdurrà la linea di colore nero (bianco su AutoCad a causa dello sfondo scuro). Dopo di che, il software richiede l’inserimento delle coordinate del punto di inizio e il punto di fine. Dopo i numeri 10, 20, 30, saranno rispettivamente richiesti x_1, y_1, z_1, le coordinate del punto di inizio, mentre dopo i numeri

11, 21, 31, saranno richieste le coordinate del punto di fine ovvero x_2, y_2, z_2. In questo caso, quindi, AutoCad genererà una linea continua dal punto di coordinate (0, 10, 0) al punto di coordinate (10, 10, 0). Dopo l'inserimento delle coordinate del secondo punto c'è uno "0" a chiudere l'entità.

Per l'entità "SOLID", ovvero solido bidimensionale pieno, si avrà un codice molto simile a quello della linea nella prima parte:

"0
SOLID
5
1
  8
1
  62
    0"

Qui, dopo lo "0" conclusivo dell'entità precedente, si avrà "SOLID" ad indicare il tipo di entità, "5" e "8" che non si possono omettere e i due "1" come identificativi dell'entità.

Come abbiamo detto prima, i numeri "1" non si possono ripetere per più entità, è quindi importante ricordare che, se il codice complessivo comprendesse davvero la linea di prima e il solid di adesso, il codice sarebbe sbagliato. Qui è sempre importante specificare il colore, quindi "62" e "0" , il quale indica il colore nero.

Dopo di che, similmente a com'era per la linea, bisognerà introdurre le coordinate di 4 punti. Se prendiamo come riferimento la *Figura 9*, dopo 10, 20, 30, bisogna mettere rispettivamente le coordinate di A(x_a, y_a, z_a), dopo 11, 21, 31, inseriamo il punto B(x_b, y_b, z_b), dopo 12, 22, 32 inseriamo il punto C(x_c, y_c, z_c) e infine dopo 13, 23, 33, inseriamo il punto D(x_d, y_d, z_d).

Figura 9. Esempio di Solid n.1

L'ordine d'inserimento dei punti è da considerare solo in quest'ordine, indipendentemente dalla forma che si vuole creare con i 4 vertici. Se invertissimo, ad esempio, le coordinate di C con le coordinate di D, il sistema non restituirà errore, ma considererà le coordinate inserite al posto di C esattamente come tali, restituendo la *Figura 10.*

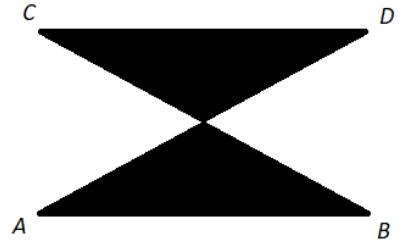

Figura 10. Esempio di Solid n.2

Questo fa comprendere che i dati da noi inseriti non vengono processati, ma presi in senso assoluto. Ciò, a mio avviso, è un vantaggio, così che non devo considerare eventuali interventi che AutoCad può fare sul mio codice modificandolo.
Per l'entità "ARC" il codice è leggermente diverso:

"ARC
5
1
8
0
10
0
20
0
30
0
40
5
50
0
51
90
0"

La stringa "ARC" definisce l'entità, "5" e "8" hanno le stesse funzioni, "1" è il numero come prima identificativo unico nel codice. A differenza di prima però, l'identificativo si ripete solo una volta mentre, dopo l'"8", c'è sempre lo "0" che indica lo spessore. Dopo lo "0", se si vuole, si può mettere "6" e il tipo di tratto "CONTINUOUS", "ACAD_ISO2W100" oppure "ACAD_ISO4W100", e il colore: "62" succeduto dal numero del colore che ci interessa. Se omessi, come in questo caso, il software considererà una linea a tratto "DaLayer" di colore nero impostata nell'header come default.
Facendo riferimento alla *Figura 11*, come prima, in seguito a 10, 20, 30, vanno inserite le coordinate di un punto che, in questo caso, è $O(x_o,y_o,z_o)$. Nella riga seguente a "40" va inserito il valore del raggio, nell'esempio 5mm. Dopo "50" va inserito il valore, in gradi, dell'angolo α e dopo "51" il valore in gradi dell'angolo β.
In questo caso α=0° e β=90° quindi l'angolo è quello in figura 11.

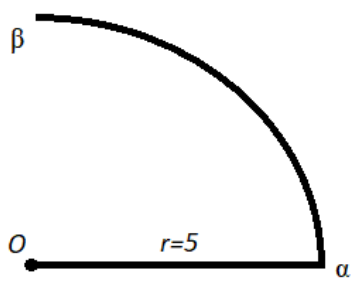

Figura 11. Esempio di Arc

Durante l'inserimento del valore dell'angolo, bisogna ricordare che per convenzione, nell'header, abbiamo definito che gli angoli vanno inseriti in senso antiorario. Se per comporre un cerchio pensiamo d'inserire lo stesso valore come angolo d'inizio e di fine, in realtà otteniamo solo un punto.
Per definire l'entità "CIRCLE" il codice è:

"CIRCLE
5
1
8
1
6
CONTINUOUS
10
635
20
1280
30
0
40
30
0"

La stringa "CIRCLE" identifica l'entità come cerchio, "5" e "8" ricoprono le stesse funzioni, i due "1" sono identificativi dell'entità e "6" introduce il tratto come in esempio "CONTINUOUS".
I numeri 10, 20, 30 in questo caso, fanno riferimento al punto $O(x_o,y_o,z_o)$ inteso come centro del cerchio e, come per l'arco, dopo "40" bisogna inserire il valore del raggio inteso in millimetri, in questo caso 30. Per chiudere l'entità, come per le altre, c'è sempre il valore "0".
Per definire l'entità "TEXT" il codice è:

"TEXT
5
1
8
1
6

CONTINUOUS
62
1
10
0
20
0
30
0
40
30
1
Testo
50
90
0”

La string “TEXT” introduce il fatto che l’entità che andremo a definire è di tipo testo, , “5” e “8” ricoprono le stesse funzioni, i due “1” sono identificativi dell’entità e “6” introduce il tratto, nell’esempio “CONTINUOUS”. Il “62”, seguito da “1”, indica che il testo in esempio sarà di colore rosso. I numeri 10, 20, 30, come sempre, richiedono delle coordinate, che facendo riferimento alla *Figura 12*, sono del punto A(x_a,y_a,z_a). Dopo il numero “40” viene inserita l’altezza in millimetri che dovranno avere le lettere, 30 nell’esempio. In seguito a “1” viene inserita la stringa di testo che si vuole inserire, AutoCad non fa distinzione tra stringhe o numeri perciò, se al posto di “Testo” avessimo messo un numero, avrebbe stampato la cifra. Il “50” e il relativo valore di seguito è facoltativo, infatti io l’ho sempre usato solo come in questo caso con il valore “90” per avere la scritta ruotata di 90° gradi in senso antiorario. Lo “0” conclude l’entità.

Figura 12. Esempio di Text

2.7 Objects

L’entità Objects definisce, di fatto, oggetti che hanno un ruolo molto simile alle entità, ma non hanno elementi di significato grafico o geometrico. Tutti gli oggetti che non sono entità o “table”, sono memorizzate in questa sezione. Questa rappresenta un heap omogeneo con ordinamento

topologico degli oggetti per proprietà, tali che i proprietari compaiono sempre prima degli oggetti che possiedono. Nei file che abbiamo creato questa sezione è omessa.

2.8 Thumbnailimage

La sezione thumbnailimage è una sezione, facoltativa, utile per informazioni relative ad abbreviazioni e convenzioni di formattazione. Ogni codice di gruppo elencato in questo riferimento è presentato da un codice di gruppo numerico e una descrizione.
Questa sezione è stata ritenuta inutile, siccome non si utilizzano abbreviazioni o convenzioni nei file delle piantine.

Ogni file .dfx deve finire con il codice:

"*ENDSEC*
0
EOF"

Indipendentemente da quale sia l'ultima sezione, essa finirà con "ENDSEC" e "0". A conclusione del file ci sarà "EOF" che sta per l'appunto per "end of file".

3. Creazione del modello di disegno per la pianta

In questo capitolo vedremo la parte realmente pratica del progetto, attraverso la quale è possibile la creazione dei file .dxf in automatico.

3.1 Scelta del linguaggio e del modus operandi

La scelta del linguaggio è stata forzata dal fatto che: il server già esistente della "TEC srl", società per la quale è stato ideato il software, è stato scritto in "PHP 7.4.21", pertanto per rendere più facile l'aggiunta di altre parti, il linguaggio scelto è stato il medesimo già utilizzato. Come IDE è stato scelto "Visual Studio Code" dato che era già stato usato proprio per i codici php durante il corso di "Progettazione di servizi web e di reti di calcolatori".
Per la creazione del codice, visto nel capitolo 2, abbiamo deciso di operare con una programmazione a oggetti, dove si definiscono gli oggetti principali all'inizio mentre nella creazione di oggetti più complessi ne uso una combinazione. Per la creazione ho usato due file: "scrittura.php" e "oggettidxf.php". Nel primo "scrittura.php" richiamo gli oggetti definiti in "oggettidxf.php" e attraverso l'uso della piattaforma XAMPP ne ho potuto verificare le operazioni volta per volta.

3.2 Definizione di oggetti per la stampa, di header e di body

Nel file "oggettidxf" è stata creata una classe denominata "DXF" nella quale si definisce, come si vede nella *Figura 13*, diverse funzioni tra cui le 3 essenziali che si chiamano "getString", "getHeaderString" , "getBodyString". Nella funzione "getString" ci limitiamo ad inizializzare la funzione "$strDXF" e a richiamare le altre due funzioni. Nella funzione "getHeaderString" si definiscono, ponendole all'inizio, tutte quelle sezioni immutabili per ogni file, ovvero: "HEADER" "TABLES" e "BLOCKS".

```php
function getString() {
    $strDXF = "";
    $strDXF .= $this->getHeaderString();
    $strDXF .= $this->getBodyString();
    return $strDXF;
}

function getHeaderString(){···
}

function getBodyString() {
    $strDXF = "";
    $strDXF .= "\nSECTION\n2\nENTITIES\n0\n";
    $strDXF .= $this->shape;
    $strDXF .= "ENDSEC\n0\nEOF";
    return $strDXF;
}
```

Figura 13. Funzioni base in "oggettidxf"

Così facendo ogni file avrà giustamente all'inizio le varie sezioni "statiche" definite e non modificabili. Nella funzione "getBodyString" si scirveranno tutte le entità, variabili per ogni disegno della sezione "ENTITIES". Anche in questa funzione c'è una parte "statica" che comprende l'inizio della sezione e la fine della sezione e con essa del file. Tra queste due parti c'è la parte invece variabile definita dalla variabile "$shape" che comprende appunto le varie entità.

```php
//STAMPA TOTALE

    if (isset($_GET['download']))
    {
        $shape->SaveFile("Disegno_Cabina2_6.dxf");
    }
    $dxfstring = $shape->getString();

    ?>
    <a href="?download">SCARICA_DXF_PIANTA</a>
</body>
</html>
```

Figura 14. Funzione di Stampa

Infine per la verifica dei file creati e della loro riuscita nel file "scrittura.php" si richiama la funzione "getString" dentro la quale sono salvati tutte le sezioni definite.

Così facendo con l'uso del modulo di Apache fornito da XAMPP si è creare una pagina HTML attraverso la quale scaricare i file salvati con "Disegno_Cabina.dxf" che una volta aperti in AutoCad mi permettevano di verificare quello che stavo scrivendo attraverso codice, in una forma grafica più comprensibile e verificabile.

3.3 Definizione delle funzioni delle entità principali

In questa parte vedremo più nel dettaglio la definizione delle funzioni per le entità modificabili e quel è stata la modalità da me ideata.

Queste funzioni saranno le uniche con una vera e propria scrittura di quello che è il file .dxf come si vede in *Figura 15*.

Una di queste, la più utilizzata e "addLine" che a differenza delle funzioni sopra citate questa ha bisogno al momento in cui viene chiamata di diverse variabili, nello specifico: "$x,$y,$z,$x2,$y2,$z2,$c,$tipo,$colore". Questo serve perché così da una stessa funzione, si ha la possibilità, attraverso uno specifico inserimento di variabili, di creare la linea come si preferisce. Prendendo il codice della linea, definito nel capitolo 2 a pagina 17, e sostituendo i numeri assoluti con delle variabili diverse a seconda della necessità, posso creare la linea del tipo e colore che preferisco della lunghezza che inserisco. Le variabili ($x,$y,$z) sono le variabili che definiscono il punto di partenza della linea, le variabili ($x2,$y2,$z2) sono le coordinate del punto in cui invece la linea finisce. $c è la variabile contatore che reiterandola ogni volta ci permette di rendere unica ogni entità rendendo quel numero, l'identificatore di quella entità.

```
function addLine($x, $y, $z, $x2, $y2, $z2,$c,$tipo,$colore) {
        if($c==197 || $c==267 || $c>=370){
            $c=$c+1000000000;
        }
    $str = "LINE\n" . "  5" . "\n" . $c."\n  8\n$c\n  6\n";
            if($tipo==0){
                $str .="CONTINUOUS\n";
            if($colore==1){
                $str .=" 62\n" . "     1\n";
            }elseif($colore==2){
                $str .=" 62\n" . "     4\n";
            }
            $str .=" 10" . "\n" . $x . "\n" . " 20" . "\n" . $y . "\n" . " 30" . "\n" . $z .
            "\n" . " 11" . "\n" . $x2 . "\n" . " 21" . "\n" . $y2 . "\n" . " 31" . "\n" . $z2 . "\n 0\n";
            }elseif($tipo==1){
            $str .= "ACAD_ISO02W100\n";
            if($colore==1){
                $str .=" 62\n" . "     1\n";
            }elseif($colore==2){
                $str .=" 62\n" . "     4\n";
            }
            $str .=" 10" . "\n" . $x . "\n" . " 20" . "\n" . $y . "\n" . " 30" . "\n" . $z . "\n" .
            " 11" . "\n" . $x2 . "\n" . " 21" . "\n" . $y2 . "\n" . " 31" . "\n" . $z2 . "\n 0\n";
            }elseif($tipo==2){
                $str .= "ACAD_ISO04W100\n";
                if($colore==1){
                    $str .=" 62\n" . "     1\n";
                }elseif($colore==2){
                    $str .=" 62\n" . "     4\n";
                }
                $str .=" 10" . "\n" . $x . "\n" . " 20" . "\n" . $y . "\n" . " 30" . "\n" . $z .
                "\n" . " 11" . "\n" . $x2 . "\n" . " 21" . "\n" . $y2 . "\n" . " 31" . "\n" . $z2 . "\n 0\n";
            }

    $this->shape .= $str;
     return $str;
}
```

@param mixed $colore

Figura 15. Definizione della funzione di linea

Il $tipo ci permette di capire se la linea interessata è: CONTINUOUS, ACAD_ISO2W100 oppure ACAD_ISO4W100 mentre $colore ci permette di definire il colore di quella linea.
Un'altra funzione è "addText" di cui vediamo il codice nella *Figura 16.*

```
function addText($x, $y, $z, $text,$c,$colore) {
    if($c==197 || $c==267 || $c>=370){
        $c=$c+1000000000;
    }
    $text = strtoupper($text);
    $c = $c+20000;
    $str = "TEXT\n" .
            "  5" . "\n" .
            $c . "\n  8\n".$c."\n  6\nCONTINUOUS\n" ;
            if($colore==1){
                $str.="    62". "\n".
              "        1"."\n";
            }
            $str.=" 10\n" .
            $x . "\n" .
            " 20\n" .
            $y . "\n" .
            " 30\n" .
            $z . "\n" .
            " 40\n" .
            "30" . "\n" .
            "  1\n" .
            $text . "\n" .
            "  0\n";
    $this->shape .= $str;

    return $str;
}
```

Figura 16. Definizione della funzione di testo

Come avevamo detto nel capitolo precedente egli ha bisogno che gli sia indicato il punto in basso a sinistra da cui far partire la scritta, punto che noi definiamo al richiamo della funzione con ($x,$y,$z), poi per poter scrivere quello di cui si ha bisogno si usa la variabile "$text" con $c si indica sempre il numero identificativo e con $colore il colore della scritta. Non è permesso di inserire i diversi tipi di testo dato che l'unico che si utilizzerà sarà il CONTINUOUS.
La funzione "addTextdxsx", nella *Figura 17* funziona in modo analogo alla funzione "addText" , infatti se guardiamo le variabili richieste non cambia nulla, bisogna soltanto ricordare che la scritta ora è in verticale (dal basso verso l'alto) e che quindi il punto da cui far partire la scritta è diverso.Dato che si è creata una funzione apposita per il testo in verticale, non è data la possibilità d'inserire l'angolo con la quale la scritta dovrà poi risultare, ho inserito sempre 90° gradi.

```
function addTextdxsx($x, $y, $z, $text,$c,$colore){
    if($c==197 || $c==267 || $c>=370){
        $c=$c+1000000000;
    }
    $text = strtoupper($text);
    $c = $c+20000;
    $str = "TEXT\n" .
        "  5" . "\n" .
        $c . "\n  8\n".$c."\n  6\nCONTINUOUS\n";
        if($colore==1){
            $str."    62". "\n".
            "        1"."\n";
        }
        $str." 10\n" .
        $x . "\n" .
        " 20\n" .
        $y . "\n" .
        " 30\n" .
        $z . "\n" .
        " 40\n" .
        //size
        "30" . "\n" .
        "  1\n" .
        $text . "\n" .
        "  50\n" .
        "90" . "\n" .
        "  0\n";
    $this->shape .= $str;
    return $str;
}
```

Figura 17. Definizione delle funzioni di testo verticale

Nella *Figura 18* si possono osservare le funzioni “addCircle” e “addArc”

.La prima funzione ha bisogno oltre che delle coordinate del centro la misura del raggio, il tipo di linea con la quale si vuole fare il cerchio e “$c” che come sempre è il numero identificatore dell’entità. La funzione che definisce l’arco invece ha bisogno come il cerchio delle coordinate del centro e del raggio, ma ha anche bisogno delle misure dell’ “angoloinizio” e dell’“angolofine”, che devono essere le misure in gradi degli angoli da cui partire e a cui finire, ricordando che ci si muove in senso antiorario.

Infine l’ultima funzione principale è “addSolid” ed è osservabile nella *Figura 19.*

Come abbiamo visto nel capitolo precedente la funzione Solid ha bisogno delle coordinate dei 4 punti, corrispettivi ai 4 vertici per riuscire a formare una figura. Pertanto la mia funzione richiede le coordinate x,y,z dei 4 punti oltre che del solito contatore per definire la funzione unicamente e del colore, nel caso in cui il Solid debba essere di colori diversi.

```
function addCircle($x, $y, $z, $raggio,$c,$tipo) {
    if($c==197 || $c==267 || $c>=370){
        $c=$c+1000000000;
    }
    $str = "CIRCLE\n" .
            "  5" . "\n" .
            $c . "\n  8\n".$c."\n  6" ;
            if($tipo==0){
            $str .= "\nCONTINUOUS\n" . " 10" . "\n" . $x . "\n" . " 20" . "\n" .
            $y . "\n" . " 30" . "\n" . $z . "\n" . " 40" . "\n" . $raggio . "\n" . " 0\n";
            }elseif($tipo==1){
            $str .="\nACAD_ISO02W100\n" . " 10" . "\n" . $x . "\n" . " 20" . "\n" . $y . "\n" . " 30" .
            "\n" . $z . "\n" . " 40" . "\n" . $raggio . "\n" . " 0\n";
            }elseif($tipo==2){
            $str ="\nACAD_ISO04W100\n" . " 10" . "\n" . $x . "\n" . " 20" . "\n" . $y . "\n" .
            " 30" . "\n" . $z . "\n" . " 40" . "\n" . $raggio . "\n" . " 0\n";
            }
    $this->shape .= $str;
    return $str;
}
function addArc($x,$y,$z,$raggio,$angoloinizio,$angolofine,$c){
    if($c==197 || $c==267 || $c>=370){
        $c=$c+1000000000;
    }
    $str = "ARC\n". "  5\n". $c ."\n". "  8\n". "0\n". "10\n". $x ."\n". "20\n".
    $y . "\n". "30\n". $z . "\n". "40\n". $raggio . "\n" .  "50\n" .
    $angoloinizio ."\n". "51\n" .$angolofine. "\n". "0\n";
    $this->shape .= $str;
    return $str;
}
```

Figura 18. Definizione delle Funzioni per il Cerchio e per l'Arco

```
function addSolid($x,$y,$z,$x1,$y1,$z1,$x2,$y2,$z2,$x3,$y3,$z3,$c,$colore){
    if($c==197 || $c==267 || $c>=370){
        $c=$c+1000000000;
    }
    $str ="SOLID\n". "5" . "\n" . $c . "\n" ;
    if($colore==1){
    $str.="    8\n$c\n    62\n        1" . "\n" ;
    }elseif($colore==2){
        $str.="    8\n$c\n    62\n        4" . "\n" ;
    }else{
        $str.="    8\n$c\n    62\n        0" . "\n" ;
    }
    $str.="    10\n". $x . "\n" . "20" . "\n" .  $y . "\n" . "30" . "\n" .
    $z . "\n" . "11" . "\n" . $x1 . "\n" . "21" . "\n" .  $y1. "\n" . "31" . "\n" .
    $z1 . "\n" . "12"."\n". $x2 . "\n" . "22" . "\n" . $y2 . "\n" . "32" . "\n" .
    $z2 . "\n". "13"."\n". $x3 . "\n" . "23" . "\n" . $y3 . "\n" . "33" . "\n" . $z3 . "\n0\n";
    $this->shape .= $str;
    return $str;
}
```

Figura 19. Definizione della funzione Solid

3.4 Utilizzo delle funzioni primarie per la realizzazione del disegno

Una volta definite e verificate le funzioni primarie, si è iniziati con la creazione delle prime vere e proprie piantine, tenendo in considerazione il fatto che si dovesse creare un modello in grado di essere modificato in base alle misure dell'impianto preso in esame. Quindi ogni misura introdotta come variabile deve essere considerata dalle funzioni in modo che se si cambia il valore, il progetto rimanga valido.

Dato che lo spazio in AutoCad è considerato come in un piano cartesiano, l'angolo in basso a sinistra del vano sarà sempre l'origine di coordinate (0,0,0) da cui far partire il disegno come si può vedere in verde nella *Figura 20*.

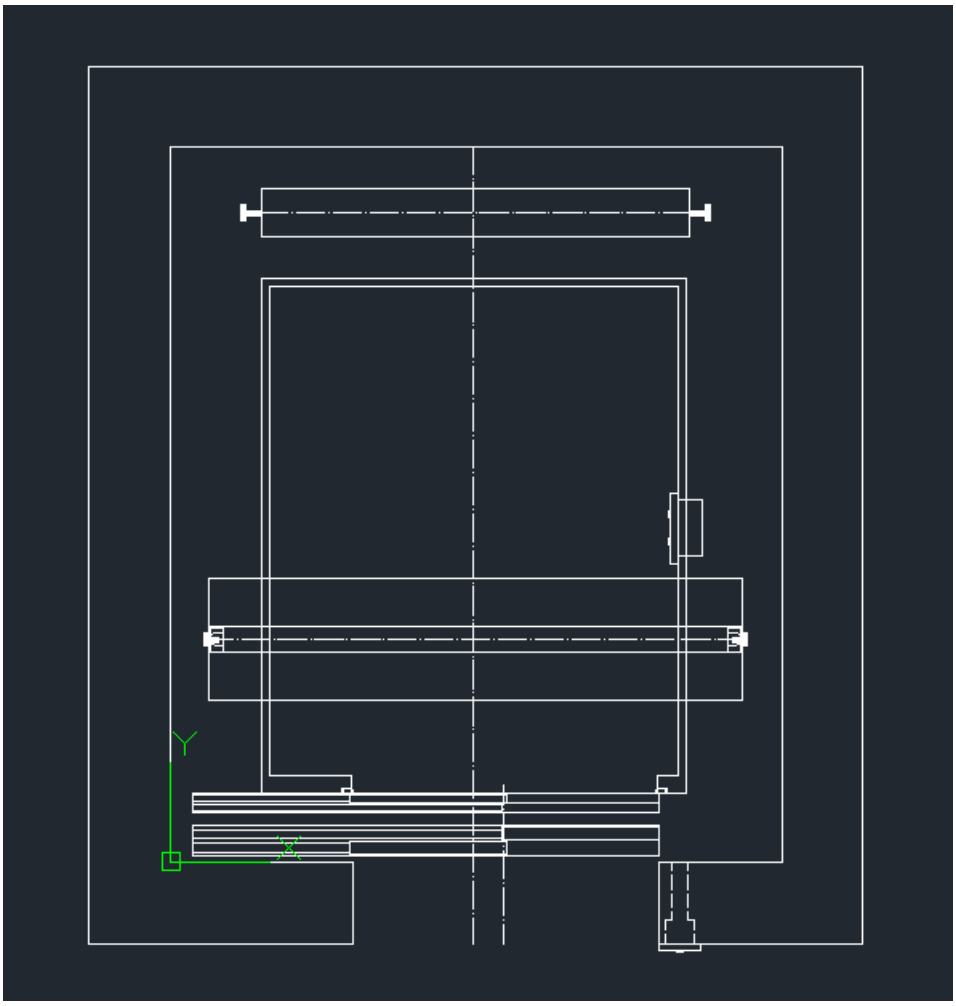

Figura 20. Disegno completo della piantina

3.4.1 Disegno del Vano

Dato che la creazione del progetto deve considerare tutti i tipi di vano ho deciso di considerare, in base al valore delle variabili, tutti gli scenari possibili. Le variabili prese in esame e chieste all'utente sono: "v_posteriore" , "v_spalletta_dx" , "v_spalletta_sx" , "v_destra" , "vc_distanza_vano_cabina" esse sono tutte precedute da un "v_" al fine di non confondere le variabili che riferiscono al vano con altre dimensioni di altre componenti della pianta ad esclusione dell'ultima, che indica la distanza tra cabina e vano quindi fa riferimento ad entrambi.
Oltre queste, inserite manualmente, ce ne sono parecchie che sono calcolate poi alla fine dell'inserimento di tutte le variabili di tutti i componenti della pianta.
Per esempio la distanza destra tra vano e cabina non è chiesta esplicitamente, ma è calcolata algebricamente sapendo i dati della larghezza del vano (v_posteriore),la larghezza della cabina e la distanza sinistra tra vano e cabina.
Nel caso specifico del vano può succedere che l'apertura dell'ascensore sia a filo su uno dei duelati eliminando così di fatto la spalletta. Sulla base di questa considerazione ho deciso, in base all'inserimento delle misure, di dividere gli scenari con delle condizioni di if come si vede in *Figura 21*.

```
380  //DISEGNO VANO
381      //variabili vano
382
383      $tipo=0;
384
385
386      //posteriore
387
388      $shape->addLine(0, $v_sinistra, 0, $v_posteriore, $v_destra, 0,$contatore,$tipo,$colore);
389      $contatore++;
390      $shape->addLine((0-$v_spessore_pareti_vano),($v_sinistra+$v_spessore_pareti_vano),0,
391      ($v_posteriore+$v_spessore_pareti_vano),($v_destra+$v_spessore_pareti_vano),0,$contatore,$tipo,$colore);
392      $contatore++;
393      $shape->addQuotadx($v_posteriore,0,$v_destra,3,"PROFONDITA' VANO ". $v_destra,$v_posteriore,$contatore);
394      $contatore=$contatore+10;
395
396 >    if($v_spalletta_dx!=0 && $v_spalletta_sx!=0){···
432 >    }elseif($v_spalletta_dx==0 && $v_spalletta_sx!=0){···
460 >    }elseif($v_spalletta_dx!=0 && $v_spalletta_sx==0){···
488 >    }elseif($v_spalletta_dx==0 && $v_spalletta_sx==0){···
510      }
```

Figura 21. Codice riassuntivo del vano

Nella *Figura 21*, possiamo osservare che: il lato posteriore del vano è sempre presente pertanto non serve metterlo in ogni condizione uguale, ma possiamo considerarlo in ogni modello a priori. La prima linea tracciata andrà quindi dal punto di coordinate (0,v_sinistra,0), dove appunto v_sinistra ha la stessa misura di v_destra e corrisponde alla misura della profondità del vano, al punto (v_posteriore, v_destra, 0) dove v_posteriore è la larghezza del vano e v_destra come prima la profondità. Dai due punti si capisce come la linea tracciata sia orizzontale. Possiamo poi

osservare che nella funzione è anche inserito il contatore, inizializzato a 1 e che dopo la scrittura della linea passa a contatore++, il tipo inizializzato a 0, che se ricordiamo com'è strutturata la funzione al paragrafo 3.3 indica una linea continua e di colore anch'esso inizializzato a zero. Se guardiamo la linea successiva, notiamo che anch'essa è una linea orizzontale, ma con lo spessore delle pareti del vano in più sulle y.
Osservando come sono strutturate queste due linee capiamo che genericamente sono sempre tracciate, ma le dimensioni cambiano al variare delle dimensioni di v_posteriore e v_destra che determinano le dimensioni del vano nella parte interna.
All'interno dei vari if ci sono aggiunte le varie linee responsabili del disegno del contorno di tutto il vano di *Figura 20*.

3.4.2 Disegno della Cabina

Come per vano, anche la cabina ha bisogno di essere disegnata pensando a tutte le possibilità di sviluppo, come si vede nel codice riportato di *Figura 22*. Questo comporta l'inserimento di diverse variabili riguardanti lo spazio di cabina, questa volta preceduti da "c_".
La parete posteriore come prima è sempre presente ed è composta da due linee, la linea interna e la linea esterna.
Analogamente a prima possiamo osservare che le coordinate di partenza possono avere diversi valori, ma non ponendo un numero per assoluto ma una variabile, al cambiare del valore cambia la posizione nel piano cartesiano. La prima linea, quella esterna della parete di cabina superiore, parte da coordinate di x dipendenti dalla distanza a sinistra tra parete e vano e finisce nelle coordinate corrispondenti al valore della somma delle variabili di vc_distanza_vano_cabina, c_larghezza_interna e lo spessore delle due pareti laterali. Così facendo non si va a limitare la grandezza della linea, ma a renderla completamente dipendente dalle dimensioni che l'utente inserisce. Per quanto invece riguarda y, lo spessore è dipendente da diversi valori anch'essi variabili, ma uguali tra il punto y_1 e y_2 così da ottenere sempre una linea orizzontale.
Come per il vano prima, all'interno dei vari if sono presenti le funzioni per il disegno delle spallette e delle pareti laterali nei vari casi.

```
226  //DISEGNO CABINA
227      //DISEGNO CABINA
228      $tipo=0;
229
230      //parete posteriore
231
232          $shape->addLine($vc_distanzasx_vano_cabina,($c_spessore_soglia+($c_spessore_pareti)+
233          $c_profondita_interna+$c_spessore_spallette_frontali),0,($vc_distanzasx_vano_cabina+
234          $c_larghezza_interna+(2*$c_spessore_pareti)),($c_spessore_soglia+$c_spessore_spallette_frontali
235          +($c_spessore_pareti)+$c_profondita_interna),
236          0,$contatore,$tipo,$colore);
237
238          $contatore++;
239
240          $shape->addLine(($vc_distanzasx_vano_cabina+$c_spessore_pareti),($c_spessore_soglia+
241          $c_spessore_spallette_frontali+$c_profondita_interna),0,($vc_distanzasx_vano_cabina+$c_spessore_pareti+
242          $c_larghezza_interna),($c_spessore_soglia+$c_spessore_spallette_frontali+$c_profondita_interna),
243          0,$contatore,$tipo,$colore);
244
245          $contatore++;
246
247          $shape->addText($vc_distanzasx_vano_cabina+$c_spessore_pareti+($c_larghezza_interna/2)-(16*30.81),
248          $s_distanza_soglia_muro+$s_spessore_soglia_di_piano+$s_distanza_soglie+$s_spessore_soglia_di_cabina+
249          $c_spessore_spallette_frontali+$c_profondita_interna-100,0,"AREA CABINA MQ ".$c_area_cabina,$contatore,1);
250
251          $contatore++;
252
253  >       //quote destra···
272  >       //quote sopra ···
295  >       if($c_spalletta_dx!=0 && $c_spalletta_sx!=0){···
334  >       }elseif($c_spalletta_dx==0 && $c_spalletta_sx!=0){···
363  >       }elseif($c_spalletta_dx!=0 && $c_spalletta_sx==0){···
393          }
```

Figura 22. Codice riassuntivo per il disegno della cabina

3.4.3 Disegno del contrappeso

Per il disegno del contrappeso ho agito diversamente, nel file “scrittura.php” ho soltanto richiamato una funzione che disegnasse il contrappeso, ma definita in “oggettidxf.php”.
Nella definizione del contrappeso ho distinto a priori il modello, come definito nel capitolo 1, o a bordiglioni o a guide. La posizione può essere laterale o a sinistra o a destra oppure posteriore. Attraverso quindi la definizione di due stringe “modello_contrappeso” e “posizione_contrappeso” ho potuto distinguere quale funzione richiamare. Nella Figura 23 qui riportata si vede com’è organizzata la distinzione posizionale nel caso del contrappeso a guide, ma è perfettamente analoga nel caso del modello a bordiglioni.
Nella *Figura 23*, si osserva come a seconda della posizione del contrappeso la funzione cambi nome, questo perché anche se parecchio simili come funzioni cambiano parecchio i dati che gli vengono passati o addirittura nel caso del contrappeso posteriore cambia l’orientamento delle guide.

```
525  //DISEGNO CONTRAPPESO
526
527
528      if($modello_contrappeso=="GUIDE"){
529          if($posizione_contrappeso=="POSTERIORE"){
530
531              $shape->addContrappesoG_Posteriore(($distanzaAttacco_Guida_Contrappeso_Orizzontale),
532              ($v_destra-$distanzaAttacco_guida_Contrappeso_verticale),0,$spessore_contrappeso,
533              $DFG_Contrappeso,$distanzaAttacco_guida_Contrappeso_verticale,$v_destra,0,$v_posteriore,$contatore);
534
535              $contatore=$contatore+10+20+30;
536
537          }elseif($posizione_contrappeso=="LATERALESX"){
538
539              $shape->addContrappesoG_Lateralesx($distanzaAttacco_Guida_Contrappeso_Orizzontale,
540              $distanzaAttacco_guida_Contrappeso_verticale,0,$spessore_contrappeso,$DFG_Contrappeso,$v_destra,0,$contatore);
541
542              $contatore=$contatore+10+20+20+10;
543
544          }elseif($posizione_contrappeso=="LATERALEDX"){
545
546              $shape->addContrappesoG_lateraledx(($v_posteriore-$distanzaAttacco_Guida_Contrappeso_Orizzontale),
547              $distanzaAttacco_guida_Contrappeso_verticale,0,$spessore_contrappeso,$DFG_Contrappeso,$v_posteriore,
548              $v_destra,$contatore);
549
550              $contatore=$contatore+10+20+10+10;
551
552          }
553
554      }elseif($modello_contrappeso=="BORDIGLIONI"){
```

Figura 23. Codice per il disegno del contrappeso in "scrittura.php"

Nella *Figura 24*, possiamo osservare come il contrappeso sia composto da due guide e 5 linee, 4 che disegnano il contorno del contrappeso e una, di tipo 2 nel codice, corrispondente alla linea di tipo "ACAD_ISO4W100", che indica il D.F.G., ovvero l'interasse tra le guide del contrappeso.
Nello specifico delle 2 guide possiamo invece osservare, dal codice della *Figura 25*, che hanno dimensioni standard e che sono composte da 2 Solid rettangolari parzialmente sovrapposti che richiamamano la forma a "T" delle guide.

Figura 24. Contrappeso posteriore

Nello specifico del disegno del contrappeso è importante sottolineare come, di fatto, il disegno sia impreciso perché: la guida non rispetta il profilo preciso che ha nella realtà, non c'è il pattino corrispondente e le guide non hanno indicate le staffe corrispondendi sulle quali poggiano. Questa imprecisione è tollerata perché il disegno, ha come fine ultimo di essere integrato alla relazione tecnica o di dare un'idea dell'impianto in generale, nelle quali è fondamentale capire la distribuzione degli spazi, le specifiche tecniche saranno poi indicate nella relazione.
Tornando in riferimento al codice, come punto da cui iniziare il disegno si passa sempre l'interno della guida sinistra nel caso di contrappeso posteriore e bassa nel caso del contrappeso laterale.
Il punto che ricevo e che richiamo come x,y,z è chiaramente condizionato dalle variabili che indico a priori nel file "scrittura.php", ma per il disegno del contrappeso questo non importa perché infatti, parto da quel punto per disegnare tutto il resto. Il disegno viene sempre fatto

indicando le coordinate dei due punti tra cui tracciare la linea e dei quattro punti che definiscono i vertici delle entità Solid che vanno poi a disegnare le guide del cotrappeso.

```
function addContrappesoG_Posteriore($x,$y,$z,$Spessore_Contrappeso,$DFG,$guide_da_alto,$v_destra,$arcata,$v_posteriore,$c){
    //quotazioni···
    $str.= $this-> addSolid(($x-12.5-12.5-12.5-12.5),($y-12.5-6.25),$z,($x-12.5-12.5-12.5),
    ($y-12.5-6.25),$z ,($x-12.5-12.5-12.5-12.5),($y+12.5+6.25),$z,($x-12.5-12.5-12.5),($y+12.5+6.25),$z,$c,0);
    $c++;

    $str .= $this->addSolid(($x-12.5-12.5-12.5),($y-6.25),$z,$x,($y-6.25), $z,($x-12.5-12.5-12.5),
    ($y+6.25),$z,$x ,($y+6.25),$z,$c,0);
    $c++;

    $str .= $this->addSolid(($x+$DFG),($y-6.25),$z,($x+$DFG+37.5),($y-6.25),$z,($x+$DFG),($y+6.25),
    $z,($x+$DFG+37.5),($y+6.25),$z,$c,0);
    $c++;

    $str .= $this->addSolid(($x+$DFG+37.5),($y+-12.5-6.25),$z, ($x+$DFG+37.5+12.5),($y+-12.5-6.25),$z,
    ($x+$DFG+37.5),($y+12.5+6.25),$z,($x+$DFG+37.5+12.5),($y+12.5+6.25),$z,$c,0);
    $c++;

    $str .= $this->addLine($x,($y),$z,($x+$DFG),($y),$z,$c,2,0);
    $c++;
    $str .= $this->addLine($x,$y-($Spessore_Contrappeso/2),$z,$x, ($y+($Spessore_Contrappeso/2)),$z,$c,0,0);
    $c++;
    $str.= $this->addLine($x,$y-($Spessore_Contrappeso/2),$z,($x+$DFG),$y-($Spessore_Contrappeso/2),$z,$c,0,0);
    $c++;
    $str.= $this->addLine($x,($y+($Spessore_Contrappeso/2)),$z,($x+$DFG),($y+($Spessore_Contrappeso/2)),$z,$c,0,0);
    $c++;
    $str.= $this->addLine(($x+$DFG),($y+($Spessore_Contrappeso/2)),$z,($x+$DFG),$y-($Spessore_Contrappeso/2),$z,$c,0,0);
    $c++;

return $str;

}
```

Figura 25. Codice contrappeso posteriore, file "oggettidxf.php"

3.4.4 Disegno del pulsante di piano, del quadro e delle linee luce

Il pulsante di piano è il pulsante posizionato sul piano, o sul telaio della soglia come in *Figura 1* o sul muro nei pressi della cabina come in *Figura 26*, che svolge la funzione di prenotare la fermata.

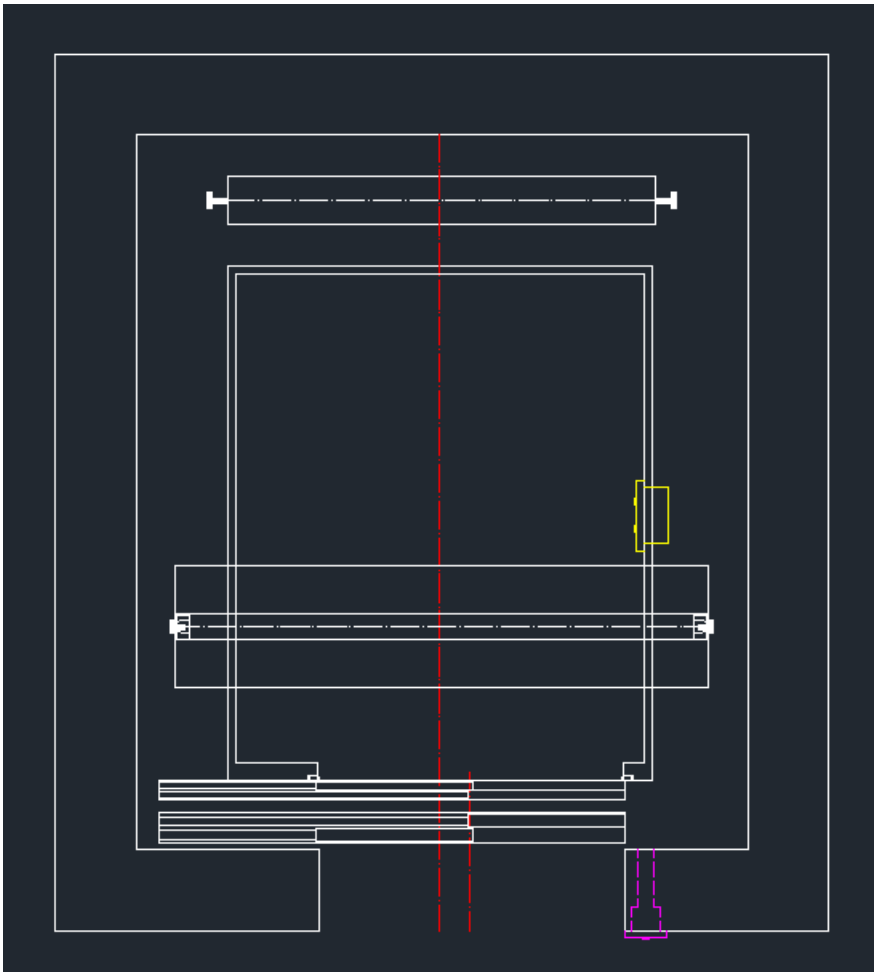

Figura 26. Evidenza linee luce, pulsantiera di piano e di cabina

Esso appunto dal punto di vista del disegno è distinto in due funzioni “addPulsantediPiano” e “addPulsanteSenzaretro” per via del fatto che se montato nel telaio il disegno comprende soltanto il pulsante e la piasta frontale, mentre se disegnato nella parete è indicato anche lo spazio destinato ai collegamenti elettrici, come si può osservare in magenta in *Figura 26*.

Quindi a seconda della posizione richiesta dall’utente cambieranno le coordinate da cui far partire il disegno, ma l’oggetto sara sempre analogo ovvero con un Solid che disegna il bottone e delle linee che vanno a definire la piastra, con l’aggiunta nel caso del pulsante di piano delle linee, di tipo 1 ovvero “ACAD_ISO2W100”, che delineano lo spazio dei componenti elettrici.

Con il disegno del quadro s’intende il vero e proprio disegno approssimativo della pulsantiera interna alla cabina, evidenziata in giallo in *Figura 26*. Per questo disegno ho agito analogamente

al cotrappeso, definendo quindi una funzione per le quatto possibili posizioni e a seconda delle coordinate faccio il disegno dove interessa l'utente.
Per invece il disegno delle linee luce s'intende la linea di tipo 2 che indica la metà della cabina e la linea che indica la metà della luce, linee rosse nella *Figura 26.* Queste ultime due linee hanno lo scopo di evidenziare quanto la cabina abbia l'apertura in centro rispetto alle pareti del vano.

3.4.5 Disegno delle fotocellule

Il disegno delle fotocellule è definito per assoluto, questo perché ai fini del disegno è importante capire dove sono locate senza effettivamente che le misure siano quelle precise di ogni modello. Nella Figura 27 si può osservare il disegno AutoCad della fotocellula di sinistra. Quella di destra è analoga, ma simmetrica. La funzione di disegno della fotocellula è singola per tutte e due, semplicemente passando all'interno della funzione anche la misura della luce, riusciamo a calcolare le coordinate anche per la fotocellula di sinistra. Questo fattore, che non vincola la fotocellula a troppi dati variabili, permette alla funzione di ricevere in input soltanto i dati del punto da cui partire, ovvero le coordinate in basso a sinistra della spalletta o parete di sinistra e la luce dell'apertura della cabina.

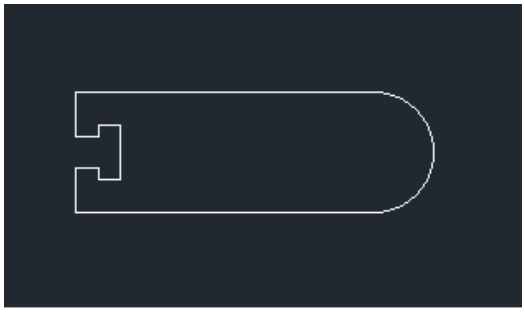

Figura 27. Disegno Fotocellula

3.4.6 Disegno della soglia

Il disegno della soglia, a differenza di tutti i precedenti richiede una della funzioni più complesse. Questo è dovuto al fatto che esistono 3 diverse combinazioni di soglia: Telescopica o Automatica, Semi-Automatica e a Battenti. Dato che alcune soglie hanno componenti in comune tra di loro, ad esempio la soglia di cabina di una soglia completamente automatica o semi-automatica è identica, ho deciso di definire le funzioni delle soglie telescopiche in di cabina e di piano differenziandole per ogni apertura. Ad esempio una soglia di cabina come in *Figura 20* sarà definita da: "addSogliaTelcab_Lateralesx" e "addSogliaTelpia_Lateralesx", ovvero il disegno della soglia di piano e il disegno della soglia di cabina. Questo mi permette in una soglia semi-automatica di riutilizzare la funzione che disegna la soglia di cabina differenziando soltanto la soglia di piano. Attraverso un sistema analogo ai precedenti quindi definisco le variabili

“Tipo_di_soglia” e “s_guide” che mi indicano rispettivamente il tipo di soglia e la direzione nella quale le porte si chiudono.

```
function addSogliaTelcab_Lateralesx($spallettasx_vano,$luce,$soglia_piano,$soglia_cabina,$distanza_soglie,
$distanza_muro_soglia,$misura_porte,$n_porte,$lunghezza_soglia,$c){
    $spessore=$soglia_piano+$soglia_cabina+$distanza_soglie+$distanza_muro_soglia;
    $larghezza_completa=$lunghezza_soglia;
    $inizio_soglia=($spallettasx_vano-($lunghezza_soglia-$luce));
    $sovrapposizione=((($misura_porte*$n_porte)-$luce)/$n_porte);//misura porte*n_porte deve essere maggiore di luce
    $distanza_scorrimento_cab=(($soglia_cabina*20)/100)/($n_porte+1);//3.333
    $spessore_porte_cab=(($soglia_cabina/$n_porte)-($distanza_scorrimento_cab+($distanza_scorrimento_cab/2)));
    $spessore_guida_porte_cab=($spessore_porte_cab-($distanza_scorrimento_cab*2));

    //disegno soglia di cabina
        //disegnocontorno···
    //disegno porte e guide
    for($i=0;$i<$n_porte;$i++){
        $distanza_sinistra=($spallettasx_vano)-($sovrapposizione)+($misura_porte*$i)-(($sovrapposizione)*$i);
        $distanza_alto=$spessore-$distanza_scorrimento_cab-($spessore_porte_cab*$i)-($distanza_scorrimento_cab*$i);
        //linea alta
        $str.=$this->addLine(($distanza_sinistra),$distanza_alto,0,($distanza_sinistra+$misura_porte)
        ,$distanza_alto,0,$c,0,0);
        $c++;
        //linea bassa
        $str.=$this->addLine(($distanza_sinistra),($distanza_alto-$spessore_porte_cab),0,($distanza_sinistra+$misura_porte)
        ,($distanza_alto-$spessore_porte_cab),0,$c,0,0);
        $c++;
        //linee laterali
        $str.=$this->addLine(($distanza_sinistra),$distanza_alto,0,($distanza_sinistra),($distanza_alto-$spessore_porte_cab),0,$c,0,0);
        $c++;
        $str.=$this->addLine(($distanza_sinistra+$misura_porte),$distanza_alto,0,($distanza_sinistra+$misura_porte)
        ,($distanza_alto-$spessore_porte_cab),0,$c,0,0);
        $c++;
        //disegno guide (2 linee e basta alta e bassa)
        //linea alta
        $str.=$this->addLine(($inizio_soglia),($distanza_alto-$distanza_scorrimento_cab),0,$distanza_sinistra,
        ($distanza_alto-$distanza_scorrimento_cab),0,$c,0,0);
        $c++;
        //linea bassa
        $str.=$this->addLine(($inizio_soglia),($distanza_alto-$distanza_scorrimento_cab-$spessore_guida_porte_cab),0,
        $distanza_sinistra,($distanza_alto-$distanza_scorrimento_cab-$spessore_guida_porte_cab),0,$c,0,0);
        $c++;
    }
```

Figura 28. Codice per il disegno della soglia telescopica di cabina con apertura a sinistra

La difficoltà principale di questa parte di disegno è che ogni soglia può avere diverse misure di lunghezza e spessore, il che non è un problema per il disegno del contorno, ma perché, più la soglia è lunga, più probabilmente avrà una chiusura telescopica comprendente numero variabile di porte. Questo è quindi un aspetto che complica notevolmente lo sviluppo di un modello generale in grado di disegnare ogni tipo di soglia.

In riferimento alla *Figura 28*, possiamo considerare lo sviluppo del disegno in tre fasi. Nella prima si calcolano delle variabili utili al disegno, ricavate da quelle definenti lo spazio nella quale è locata la soglia. Nella seconda, nascosta sotto “//disegno contorno”, si disegna il perimetro della soglia in modo analogo a come fatto fino ad adesso anche per le altre funzioni. Nella terza ed ultima invece si vanno a disegnare le singole porte e le corrispondenti guide di scorrimento. Per quest’ultima parte ho deciso di disegnare singolarmente ogni porta con relativa guida all’interno di un for che parte da 0 e disegna fino al numero di porte definito dall’utente. Questo metodo permette di disegnare quindi un numero di porte variabile in modo ordinato definendo gli spazi di

distanza tra una porta e l'altra nella prima fase e aumentando le dimensioni ad ogni iterazione del ciclo.

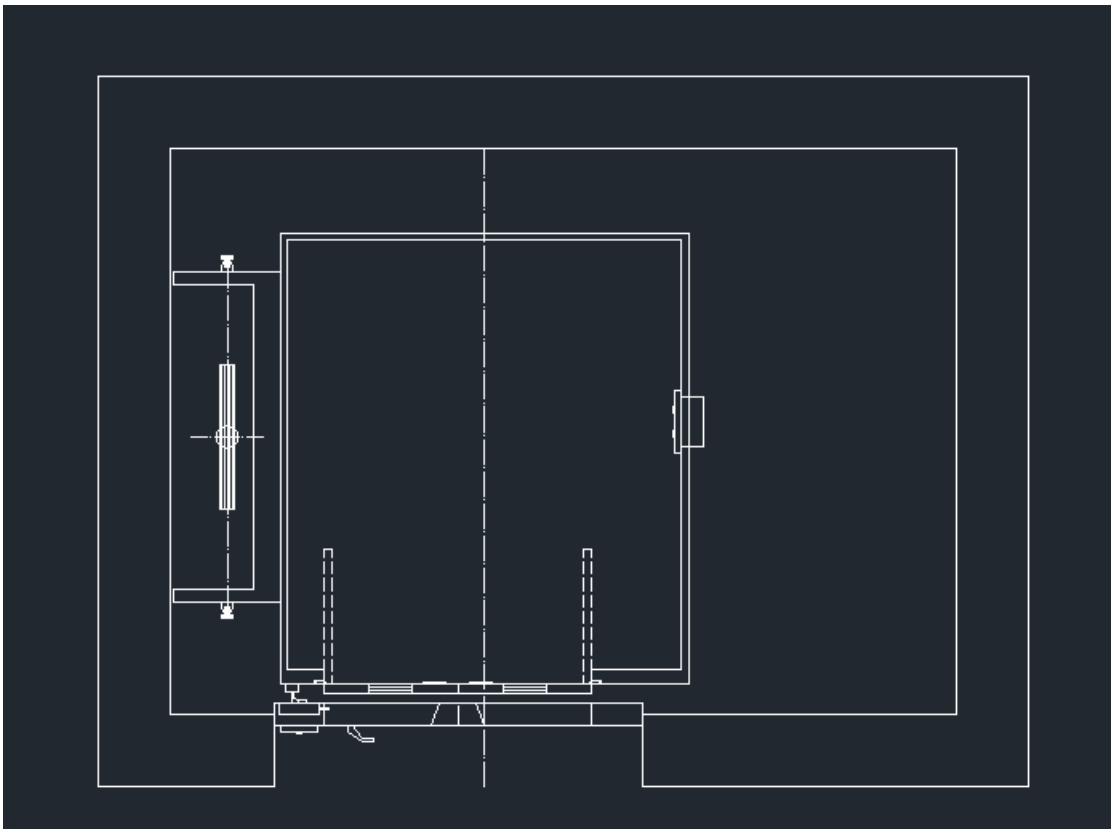

Figura 29. Disegno soglia a battenti

Nella *Figura 29* si può invece osservare un esempio di soglia con porte a battenti. Nel caso di questo ultimo disegno, le funzioni, dalla parte del codice sono sempre divise tra funzione di piano e di cabina, ma non avendo da considerare un variabile numero di porte, il disegno è sempre il medesimo, variano solo le misure di lunghezza delle porte.
Sempre riguardati gli spazi di soglia ci sono le funzioni riguardanti i riporti di soglia variabili. Questi ultimi sono dei prolungamenti variabili che compensano le differenze di distanza tra vano e cabina nei diversi piani.

3.4.7 Disegno dell'arcata

Il disegno dell'arcata è definito attraverso la creazione di 4 funzioni distinte, rispettivamente: l'arcata ad anello, l'arcata a sedia e il pistone idraulico orizzontale e verticale. La combinazione delle suddette o l'uso singolo permettono di disegnare tutti i casi possibili di arcate usate oggigiorno negli impianti di sollevamento sia oleodinamici sia elettrici.

Analogamente al contrappeso noi indichiamo solamente le guide sulle quali scorre l'arcata, ma non le staffe sulle quali sono montate, perché superflui al fine dei nostri disegni.
Nella *Figura 30* possiamo osservare un esempio di arcata a sedia, in rosso, con pistone in taglia di colore giallo.

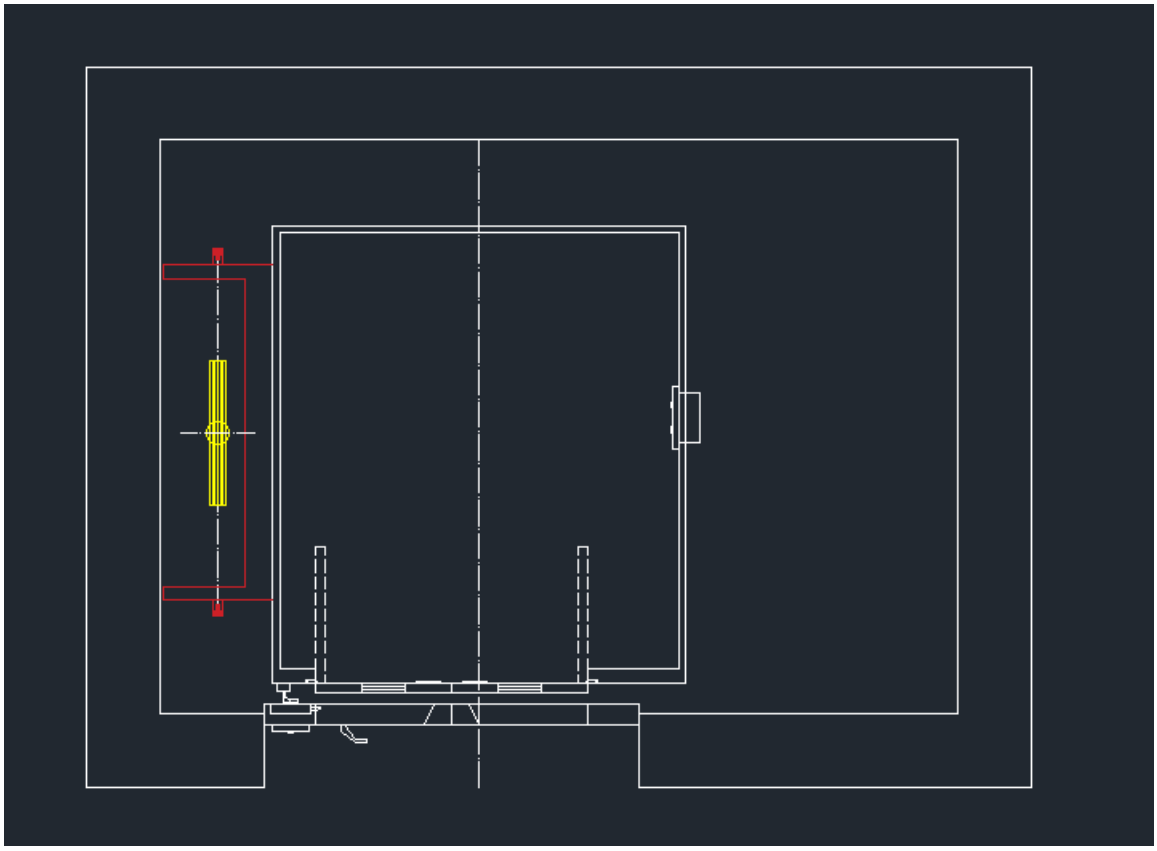

Figura 30. Disegno arcata a sedia con pistone singolo

La funzione dell'arcata comprende le guide definite come "solid" e diverse linee che compongono oltre che l'arcata in sé, anche i due diversi pattini e l'arcata vera e propria.
Il disegno del pistone, di dimensioni standard, varia unicamente in base alla posizione dell'arcata o, nel caso di un'arcata ad anello, per via di un doppio pistone. La dimensione dell'arcata a sedia varia in base alle scelte dell'utente di modello e grandezza. Se avessimo considerato un impianto senza pistone, il disegno della singola arcata sarebbe rimasto invariato, quindi solo la parte in rosso, ma con l'aggiunta di un contrappeso essenziale in un impianto elettrico.
L'arcata ad anello, come dicevamo nel primo capitolo, abbraccia la cabina ed è caratterizzata da guide su due lati opposti della cabina, come si osserva dalla *Figura 31* in rosso. Come per la *Figura 30* i pistoni possono essere aggiunti anche nel caso di un'arcata ad anello, solitamente due in taglia o uno in centro

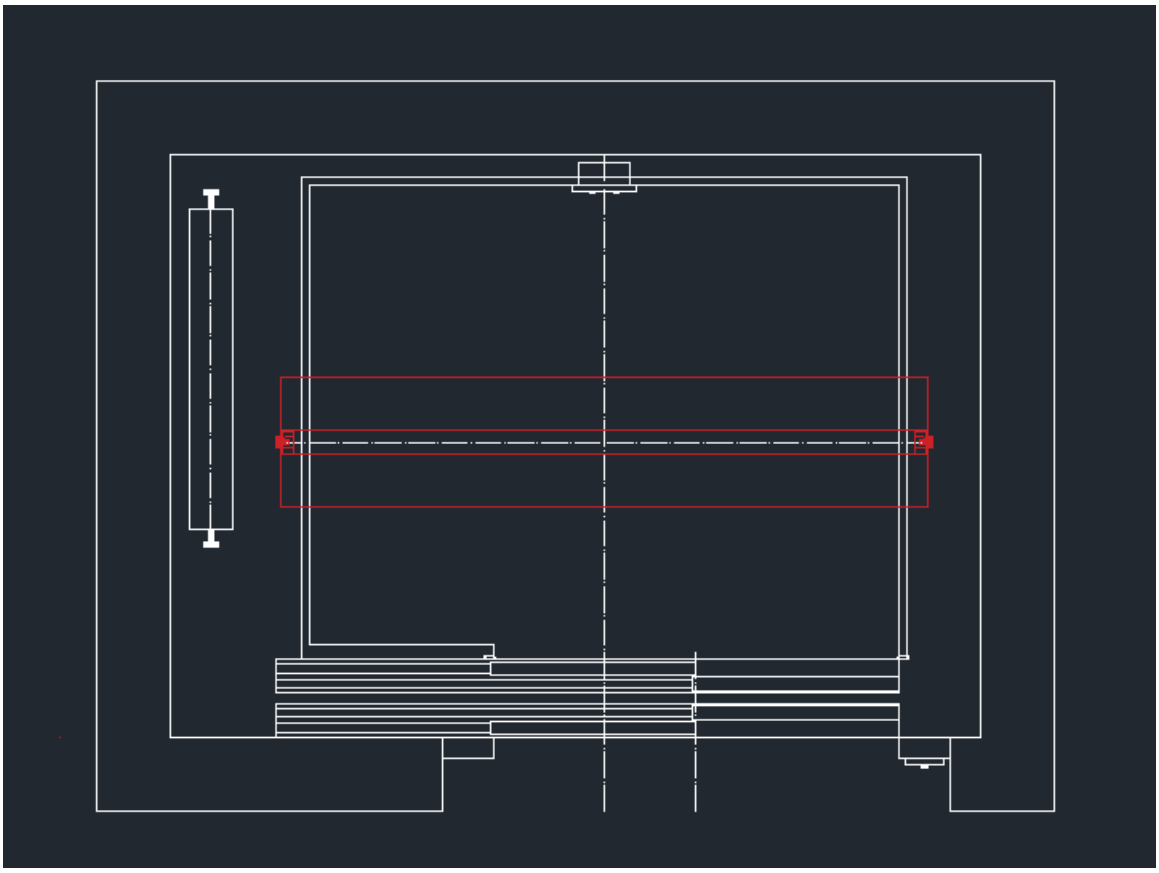

Figura 31. Disegno arcata ad anello

3.4.8 Disegno delle quote

Il disegno delle quote, contrariamente a quanto si potesse pensare non ha un'entità a se stante, ma è la combinazione di "line", "solid" e "text". Per ottenere la freccia caratteristica della quota è bastato sovrapporre due dei quattro punti che definiscono il solid, così da ottenere un triangolo. Per le scritte la difficoltà è stata quella di porre la scritta perfettamente in centro alla quota, perché l'entità "text" richiede il punto da cui far partire la scritta, ma nella realtà dei disegni è importante che essa sia posta in centro. Quindi sperimentalmente sono state misurate le lunghezze delle diverse lettere e numeri, poi una volta fatta la media, in base alla lunghezza della stringa che si andrà a scrivere, è stato possibile calcolare la lunghezza in mm della scritta e di conseguenza calcolare il punto preciso per far partire la scritta. Nel caso invece di quote molto piccole, la scritta è stata posta lateralmente, come nel caso della *Figura 32* per lo spessore delle pareti e della distanza tra le soglie. Si può osservare un esempio di disegno completamente quotato nella *Figura 32.*

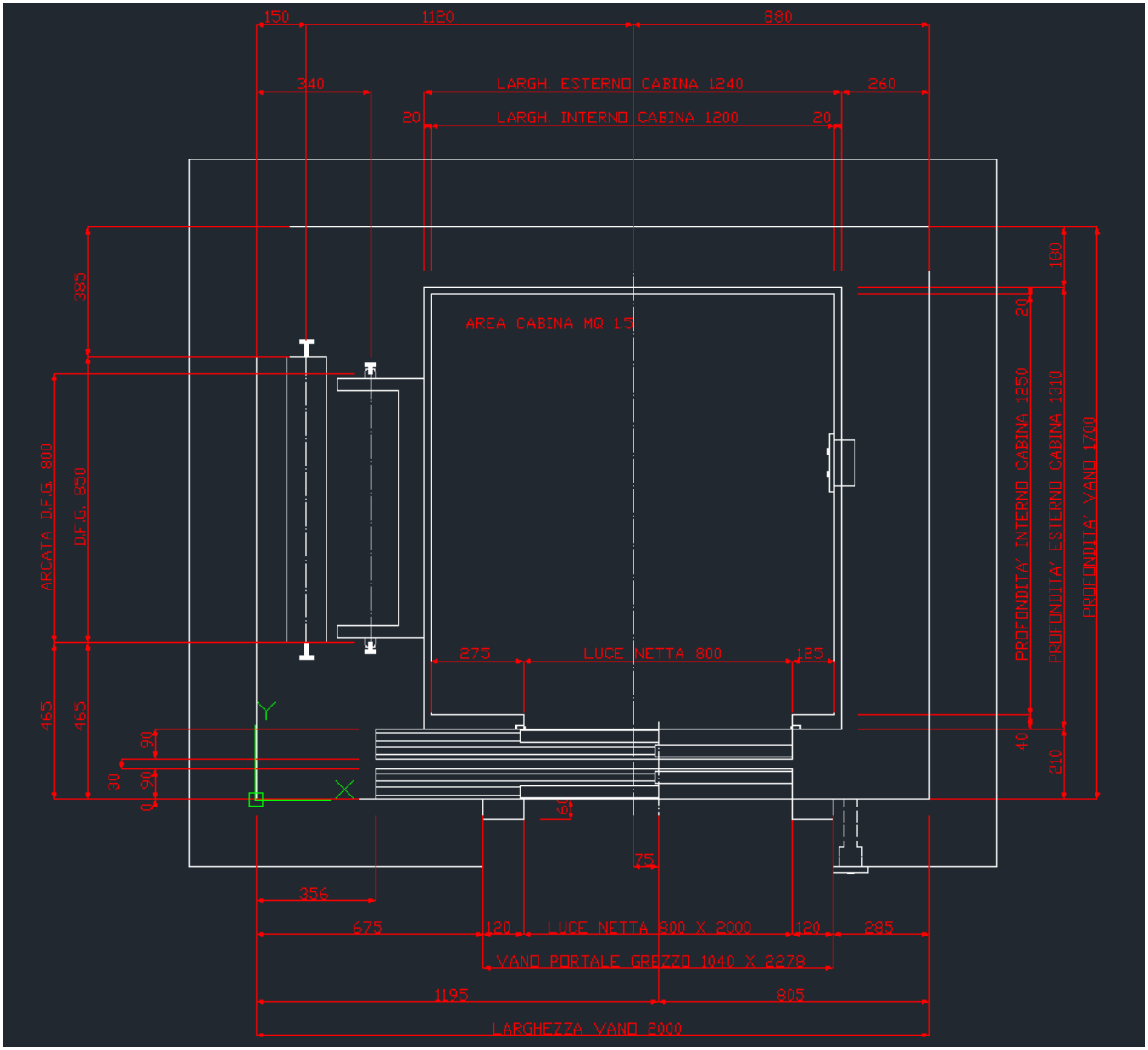

Figura 32. Disegno pianta di impianto di sollevamento completo di quote

Dal punto di vista del codice, la funzione per il disegno della quota è diversa a seconda della direzione nella quale la si vuole sviluppare, esisteranno quindi quattro funzioni nominate: “addQuotasx”, “addQuotadx”, “addQuotaup” e “addQuotadw”.

La funzione quota, il quale codice è osservabile nella *Figura 33*, presenta una variabile chiamata livello, che permette alle quote di non sovrapporsi. Nel disegno di *Figura 32* nello specifico possiamo osservare ad esempio sei livelli di quote verso il basso. Ogni livello si riserva ad una determinata quota, nel caso di quote verso il basso, il livello 0 è riservato alle quote che definiscono la differenza tra il centro di cabina e il centro della luce, in questo caso “75”. Il livello 1 è stato riservato alla distanza tra l’incasso della soglia e il vano. Il livello 2 è riservato alle misure delle spallette degli eventuali telai e della luce. Il livello 3 è riservato alle misure del vano grezzo. Il livello 4 serve ad evidenziare la distanza laterale tra il vano e il centro della luce. Il livello 5 invece è riservato alla larghezza del vano.

Questa precisa divisione in divelli è riservata in modo specifico ad ogni entità. Questa specifica divisione permette di non sovrapporre le linee perché, ad esempio, se non fosse presente il telaio, il livello 3 inferiore sarebbe vuoto.

In questo caso specifico, il livello 3 superiore, è riservato alle dimensioni del contrappeso che però essendo posto a sinistra non esistono.

```
1765    function addQuotasx($distanza_sinistra,$distanza_basso,$misura_quotare,$livello,$testo,$c){
1766
1767        $distanza_livelli=100;
1768        $distanza_iniziale=50;
1769        $spessore_freccia=10;
1770        $lunghezza_freccia=20;
1771
1772            //linee
1773            $str=$this->addLine($distanza_sinistra-$distanza_iniziale,$distanza_basso,0,
1774            0-200-$distanza_livelli*$livello, $distanza_basso,0,$c,0,1);
1775            $c++;
1776            $str.=$this->addLine($distanza_sinistra-$distanza_iniziale,$distanza_basso+$misura_quotare,0,
1777            0-200-$distanza_livelli*$livello,$distanza_basso+$misura_quotare,0,$c,0,1);
1778            $c++;
1779            $str.=$this->addLine(0-200-$distanza_livelli*$livello,$distanza_basso,0,0-200-$distanza_livelli*$livello,
1780            $distanza_basso+$misura_quotare,0,$c,0,1);
1781            $c++;
1782            //freccia bassa
1783            $str.=$this->addSolid(0-200-$distanza_livelli*$livello,$distanza_basso,0,0-200-$distanza_livelli*$livello,
1784            $distanza_basso,0,0-200-$distanza_livelli*$livello-($spessore_freccia/2),$distanza_basso+$lunghezza_freccia,
1785            0,0-200-$distanza_livelli*$livello+($spessore_freccia/2),$distanza_basso+$lunghezza_freccia,0,$c,1);
1786            $c++;
1787            //freccia alta
1788            $str.=$this->addSolid(0-200-$distanza_livelli*$livello-($spessore_freccia/2),$distanza_basso+$misura_quotare-$lunghezza_freccia,
1789            0,0-200-$distanza_livelli*$livello+($spessore_freccia/2),$distanza_basso+$misura_quotare-$lunghezza_freccia,0,
1790            0-200-$distanza_livelli*$livello,$distanza_basso+$misura_quotare,0,0-200-$distanza_livelli*$livello,$distanza_basso+$misura_quotare,0,$c,1);
1791            $c++;
1792
1793            $lung_testo=strlen($testo)*30.81;
1794            $y=$distanza_basso+$lunghezza_freccia+($misura_quotare/2)-($lung_testo/2);
1795            if($misura_quotare<50){
1796                $y=$distanza_basso-$lung_testo;
1797            }
1798
1799            $str.=$this->addTextdxsx(0-200-$distanza_livelli*$livello+($spessore_freccia/2)-15,$y,0,$testo,$c,1);
1800            $c++;
1801
1802        return $str;
1803    }
```

Figura 33. Funzione per il disegno della quota sinistra

4. Creazione parte front-end e collegamento al database

La creazione dell'interfaccia utente è stata eseguita in linguaggio HTML e CSS così da poter utilizzare lo stesso stile dell'attuale web app della TEC.

4.1 Index.html

L'interfaccia segue le stesse regole per l'inserimento dati già esistente nell'altra sezione della web app, per la creazione dei libretti per le certificazioni degli impianti.
Tutti i dati sono inseriti attraverso dei form con metodo post che alla pressione del bottone "Crea disegno AutoCad" invia tutti i dati al file "scrittura.php" che crea il disegno in formato dxf scaricandolo in automatico nei download. Gli input, o chiedono direttamente la misura, oppure, attraverso un menù a tendina, ti permettono di scegliere tra diverse opzioni predefinite, vedi *Figura 34*.

```
<hr class="my-5">
 <h3 class="text-center mt-5 mb-3">
Soglia</h3>

<div class="form-row">
    <div class="form-group col-md-6">
        <label for="soglia_tipo">Tipo e posizione</label>
        <select name="soglia_tipo" id="soglia_tipo" class="form-control" required>
            <option selected value="" disabled>SCEGLI</option>
            <option value="telescopica_____centrale">Telescopica con apertura centrale</option>
            <option value="telescopica_____lateralesx">Telescopica con apertura a sinistra</option>
            <option value="telescopica_____lateraledx">Telescopica con apertura a destra</option>
            <option value="semiautomatica___centrale">Semi-Automatica con apertura centrale</option>
            <option value="semiautomatica___lateralesx">Semi-Automatica con apertura a sinistra</option>
            <option value="semiautomatica___lateraledx">Semi-Automatica con apertura a destra</option>
            <option value="battenti">Battenti</option>
        </select>
    </div>
    <div class="form-group col-md-3">
        <label for="soglia_numero_porte_piano">N° porte piano</label>
        <input type="number" name="soglia_numero_porte_piano" class="form-control" value=0 >
    </div>
```

Figura 34. I due esempi d'inserimento dati

Questa seconda opzione è stata usata nel caso in cui le scelte dell'utente sono limitate, come per esempio: la scelta del modello dell'arcata, la posizione dell'arcata, la posizione del pulsante di piano o il modello e la posizione della soglia come nell'esempio all'inizio di *Figura 34*.
Nel caso invece in cui viene richiesto l'inserimento di un vero e proprio numero, come nel secondo " " di *Figura 34*, l'inserimento del dato sotto la voce "N° porte di piano", può essere definito con un valore di default attraverso la voce "value=0".

Spesso sono stati inseriti dei valori di default che corrispondessero ai valori più comuni inseriti in quel campo. Nel caso ad esempio dell'altezza del telaio, il valore di default è 2000mm perché secondo le normative vigenti deve essere alto almeno 2 metri e solitamente viene lasciato tale. Questo è stato fatto al fine di rendere più veloce la compilazione dei dati.

4.2 CSS

Per la creazione dello stile è stato fatto affidamento alle già usate, nella restante web app, librerie bootstrap[2]. Esse hanno permesso l'inserimento dei dati attraverso dei riquadri uniformi nello stile e delle tendine con ombre e linee di selezione evidenziate. La classe CSS usata è "form-control", come si vede l'inserimento in *Figura 34* e il risultato grafico in *Figura 35*.
Inoltre è stato possibile definire dei margini a destra e sinistra così da rendere più bella l'impaginazione e di separare le varie categorie attraverso una linea. Vedi *Figura 35*.

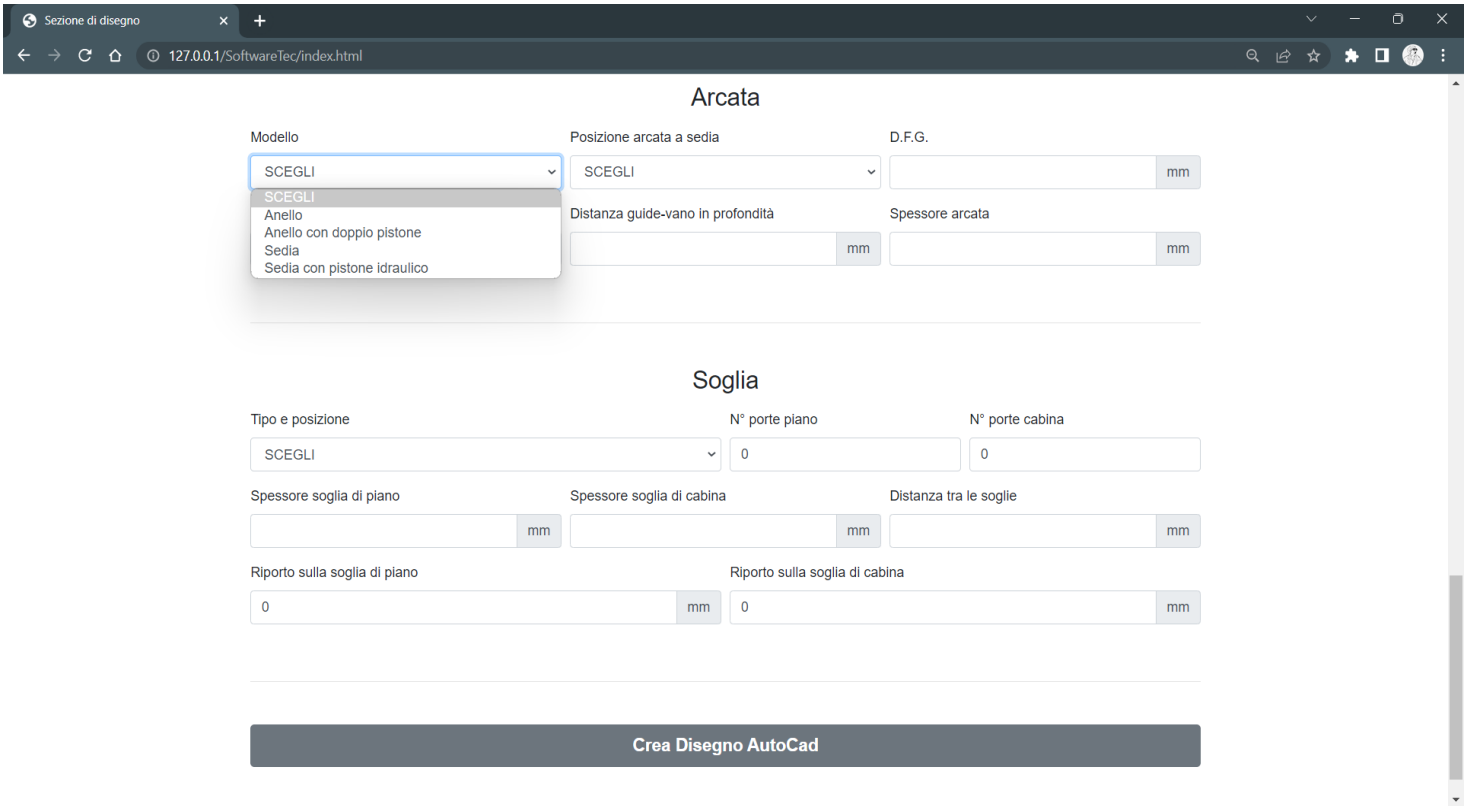

Figura 35. Immagine dell'interfaccia utente

Ogni riga dell'interfaccia utente, definita all'interno di " …. " dobbiamo immaginarla divisa da 12 colonne che permettono di gestire lo spazio.
Quindi per fare ad esempio un input di metà pagina, all'interno della classe "form-row" va richiamata la classe "form-group col-md-6" all'interno di un ulteriore div, che permette di occupare 6 delle dodici colonne. Se nella restante metà si volessero fare altri 2 input va utilizzata la classe "form-group col-md-3" altre 2 volte così da dividere lo spazio in 6+3+3. Vedi *Figura 35*, la prima riga nella sezione "Soglia" rispetta questa divisione presa in esempio.

4.3 Connessione al database

La connessione al database è indispensabile al fine di poter svolgere il disegno in diversi momenti.
Nella pagina principale del software, a monte di questa pagina: "index.html", vengono mostrati i rilievi già inseriti con corrispettivi disegni e libretti, attraverso una finestra con le specifiche. Vedi *Figura 36.*
Il nome del collaudatore viene dato in automatico al Login sulla pagina, mentre l'ID della scheda di collaudo viene fornito in modo progressivo da 0.

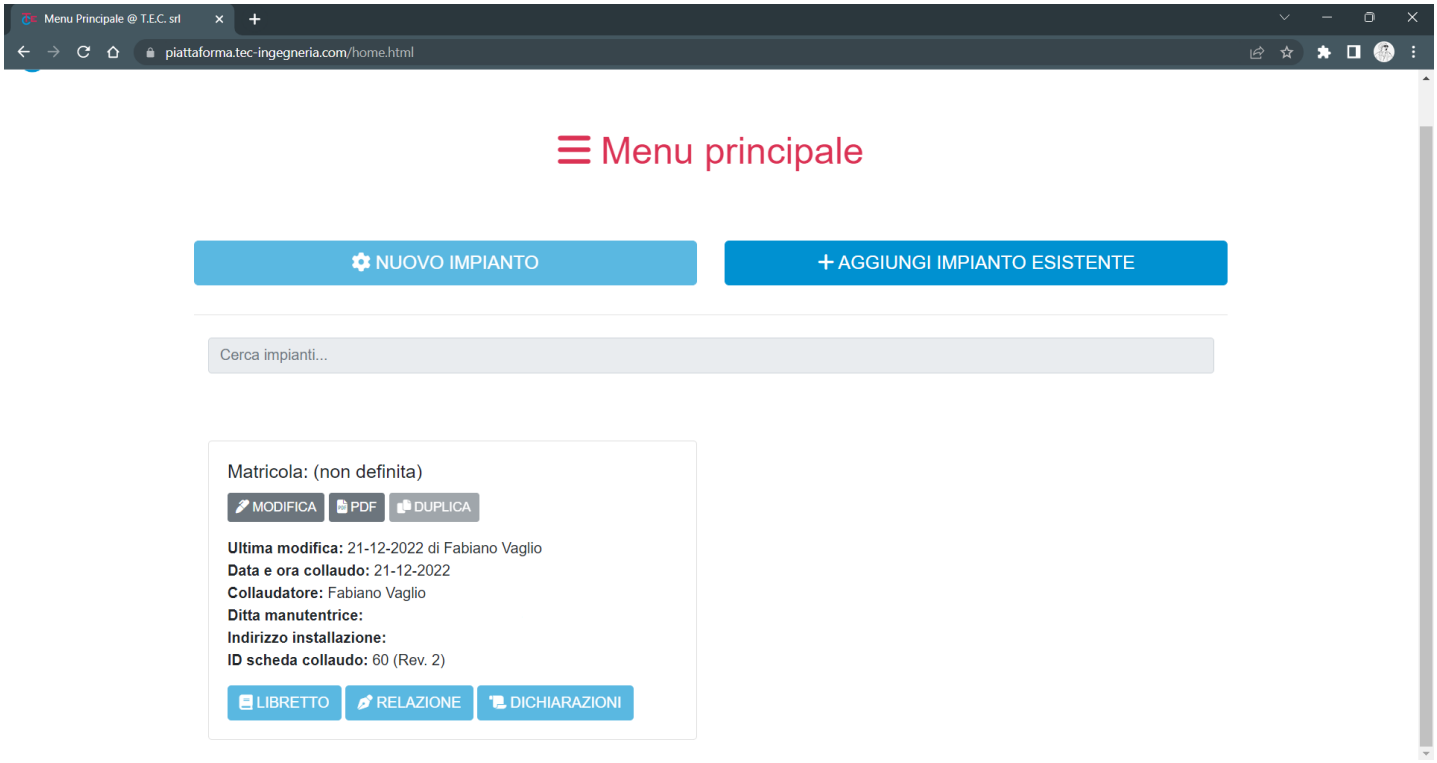

Figura 36. Interfaccia principale

Tutti i restanti dati vengono inseriti manualmente nel database denominato: “datiNuoviImpianti”, attraverso il codice in *Figura 37* , così che, all’inserimento dei dati nella pagina creata e spiegata in questa tesi, si ha memoria di tutti gli impianti che vengono disegnati o che, attraverso altre pagine ne viene fatta la relazione tecnica.

```php
$conn = new mysqli("localhost", "root","","test");

if($conn->connect_error){
    die("Connessione fallita".$conn->connect_error);

}


$sql = "INSERT INTO datiNuoviImpianti (…
VALUES (…
if (mysqli_query($conn, $sql)) {
echo "New record created successfully";
} else {
echo "Error: " . $sql . "<br>" . $conn->error;
}
$conn->close();
```

Figura 37. Codice di connessione al Database

5. Esempi di disegni sviluppati

Qui di seguito sono stati riportati diversi disegni creati su impianti reali di Torino, al fine di verificare la correttezza dei disegni in tutte le loro particolarità.

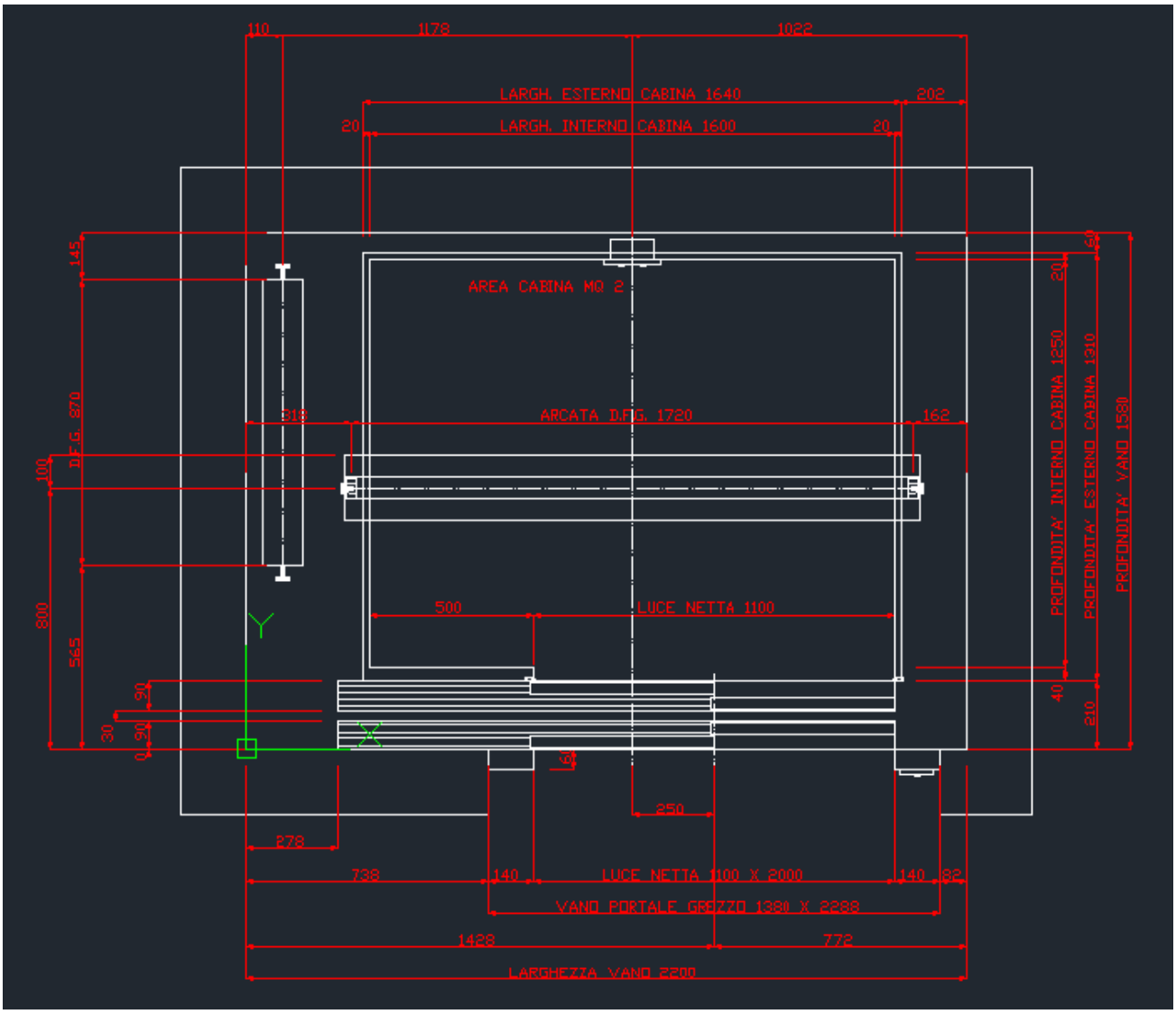

Figura 38

Nella *Figura 38* qui riportata si può notare una cabina, di un impianto elettrico con arcata ad anello e contrappeso a guide. La soglia è di tipo telescopica con apertura a sinistra e con un telaio comprendente il pulsante di piano.

Lo stesso impianto è stato ridisegnato a *Figura 39* con una proposta più economica.

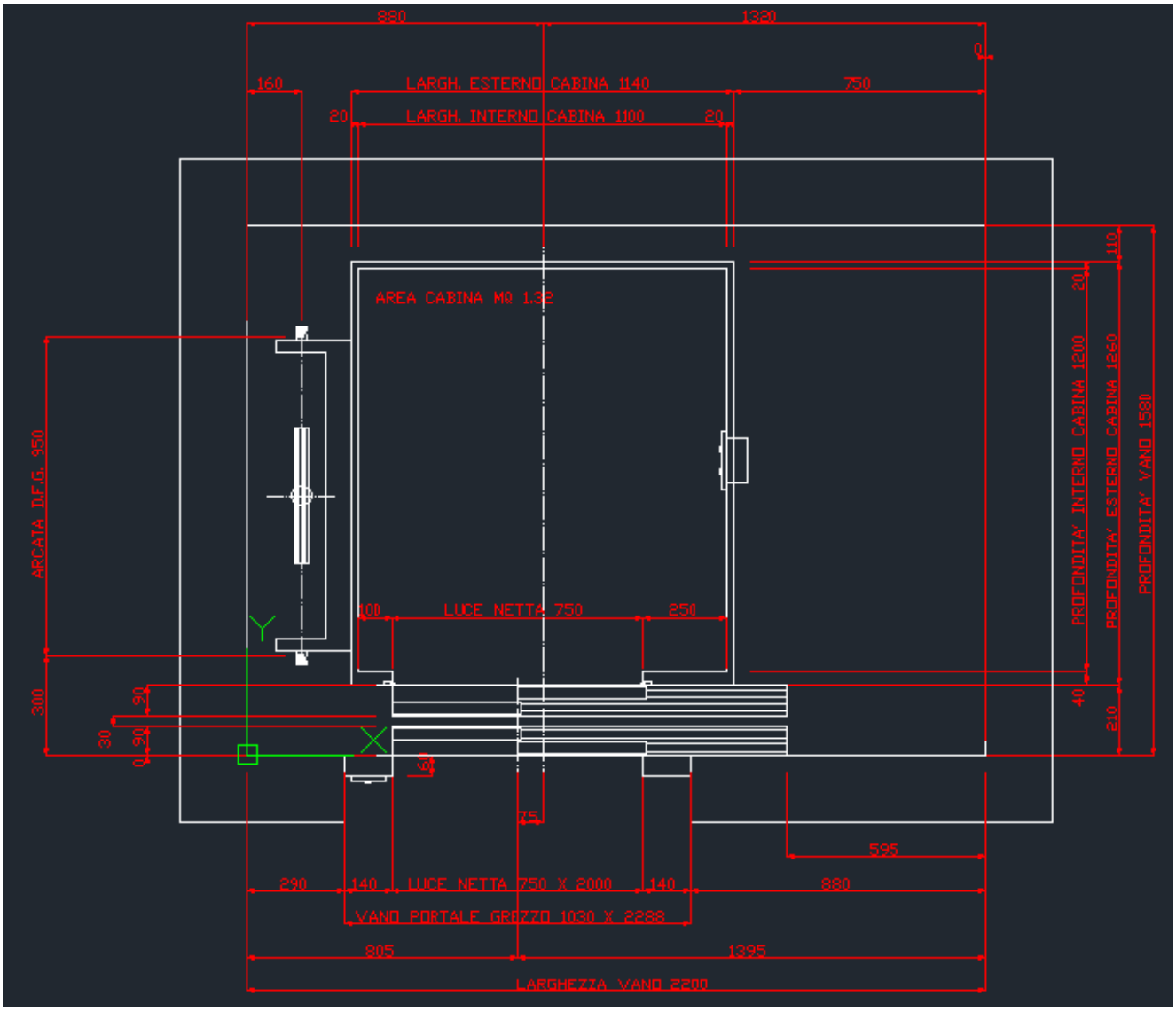

Figura 39

Come si può osservare le dimensioni del vano sono le stesse, ma qui è proposta una soluzione idraulica con arcata a sedia a sinistra. Di conseguenza la soglia è stata spostata a destra per questione di spazi. L'arcata in questo caso non può essere troppo distante dal vano.

Prediligendo un impianto oleodinamico, si sono dovute sacrificare le dimensioni interne della cabina, altrimenti non sarebbe stato sufficiente un solo pistone in taglia per il sollevamento.

La seconda soluzione è più economica sia per via della dimensioni che per il tipo d'impianto scelto.

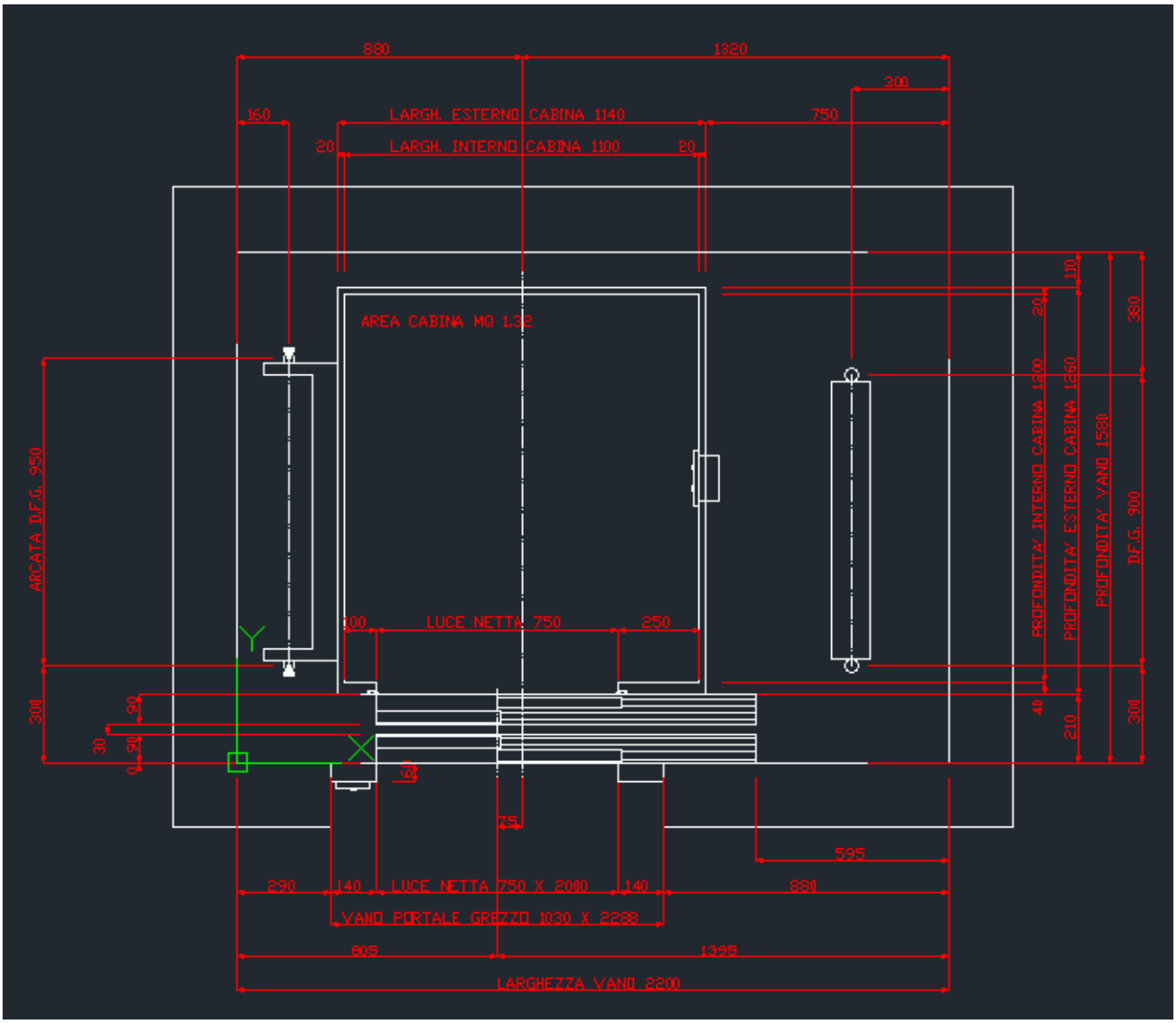

Figura 40

Una terza soluzione è proposta a fini esplicativi. La *Figura 40* infatti è analoga alla *Figura 39* rappresenta il medesimo impianto, ma con un'arcata a sedia e un contrappeso a bordigioni

La soluzione del contrappeso a bordiglioni non è più utilizzata, ma ai fini del progetto è stata inserita la possibilità di disegnarli, nel caso in cui servissero disegni di impianti anche soltanto riparati o d'impianti di cui si è smarrito il libretto.

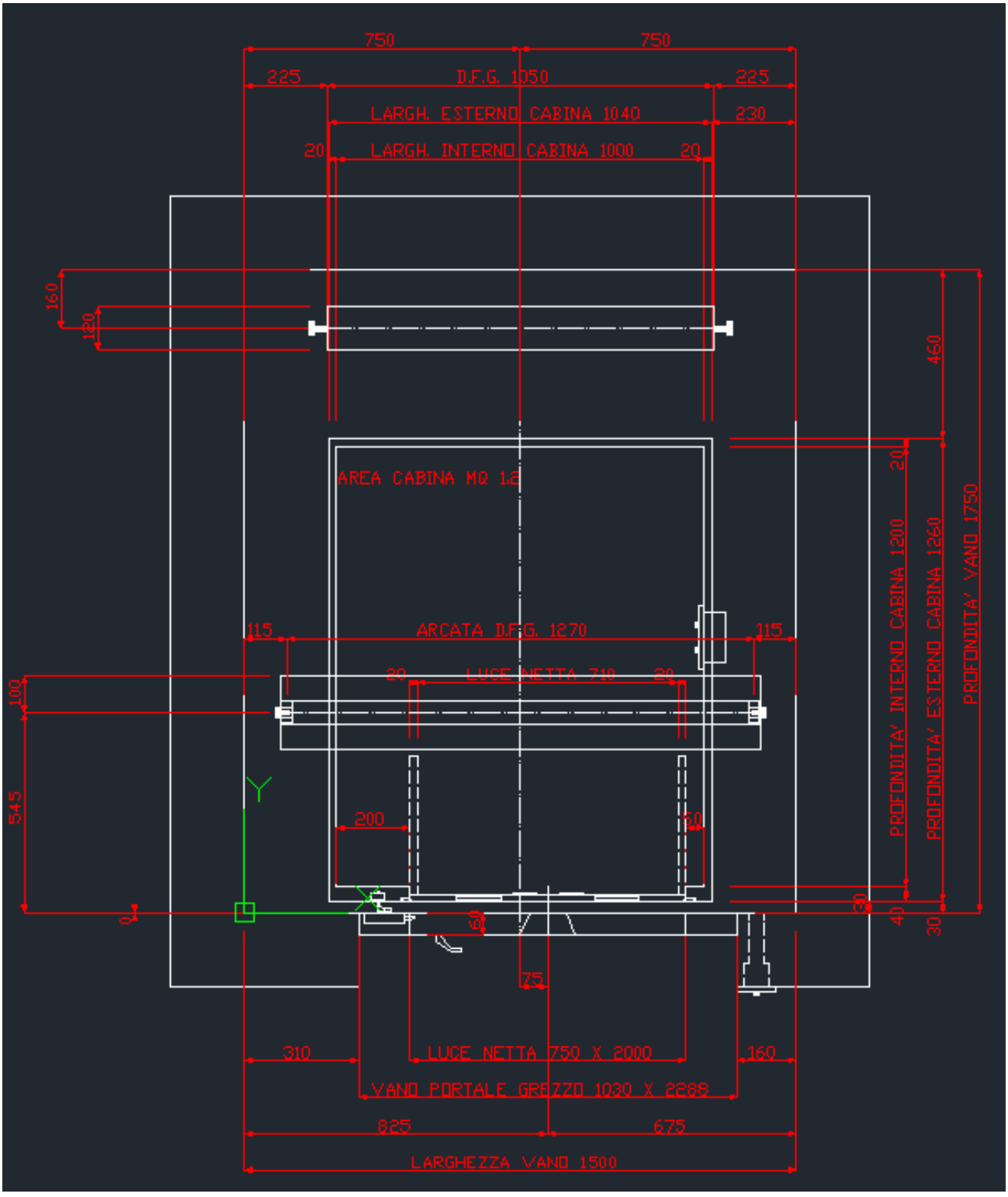

Figura 41

Nella *Figura 41*, possiamo osservare un impianto originale prima dell'intervento di ammodernamento. Nello specifico l'ammodernamento prevedeva di modificare la soglia al fine di recuperare spazio all'interno della cabina.

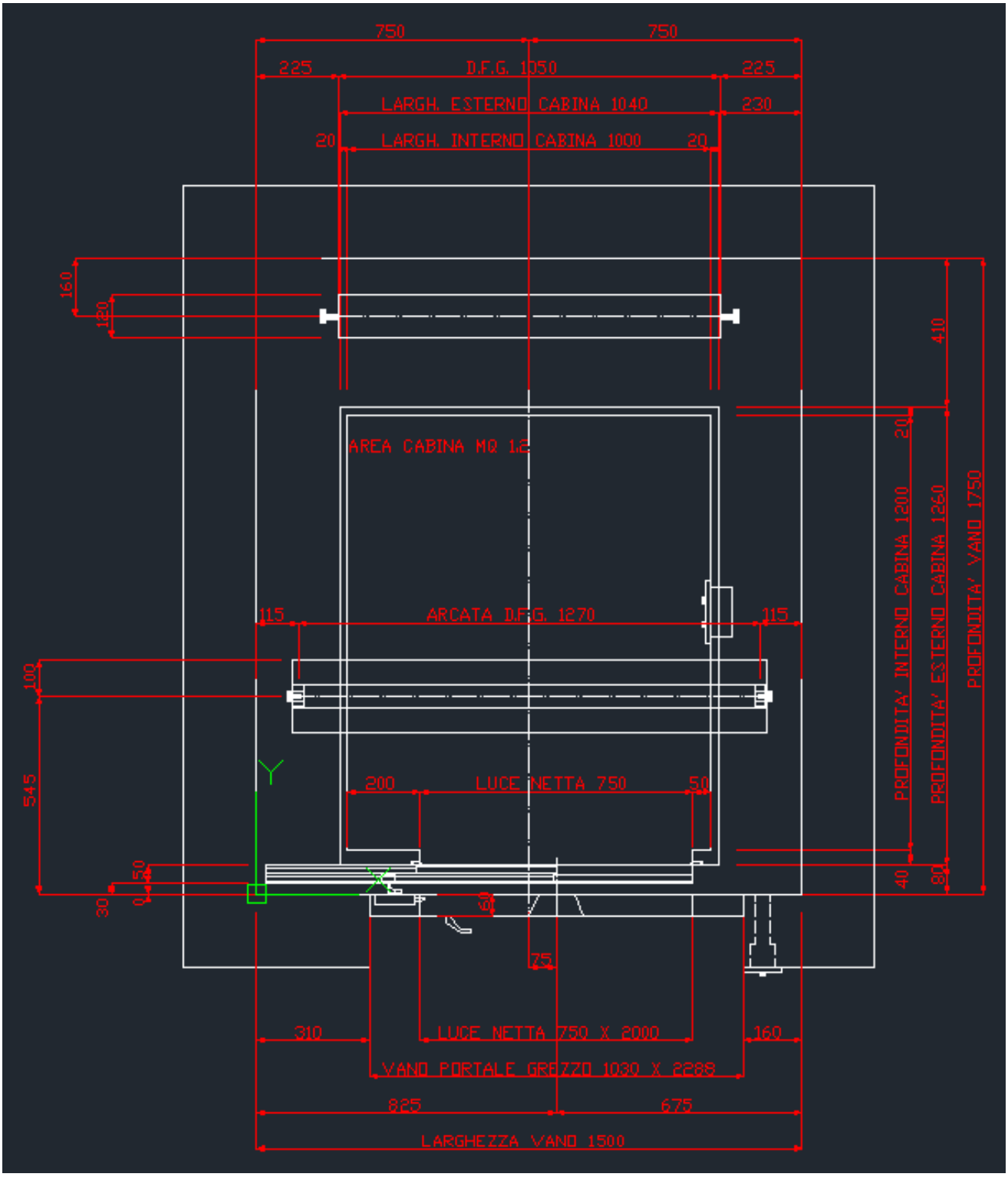

Figura 42

Nella *Figura 42* possiamo quindi osservare un'idea di ammodernamento che consiste nel modificare esclusivamente la soglia di cabina, con una soglia telescopica, così da recuperare lo spazio che occuperebbero le porte di cabina nell'apertura senza modificare l'ingresso ad ogni piano.

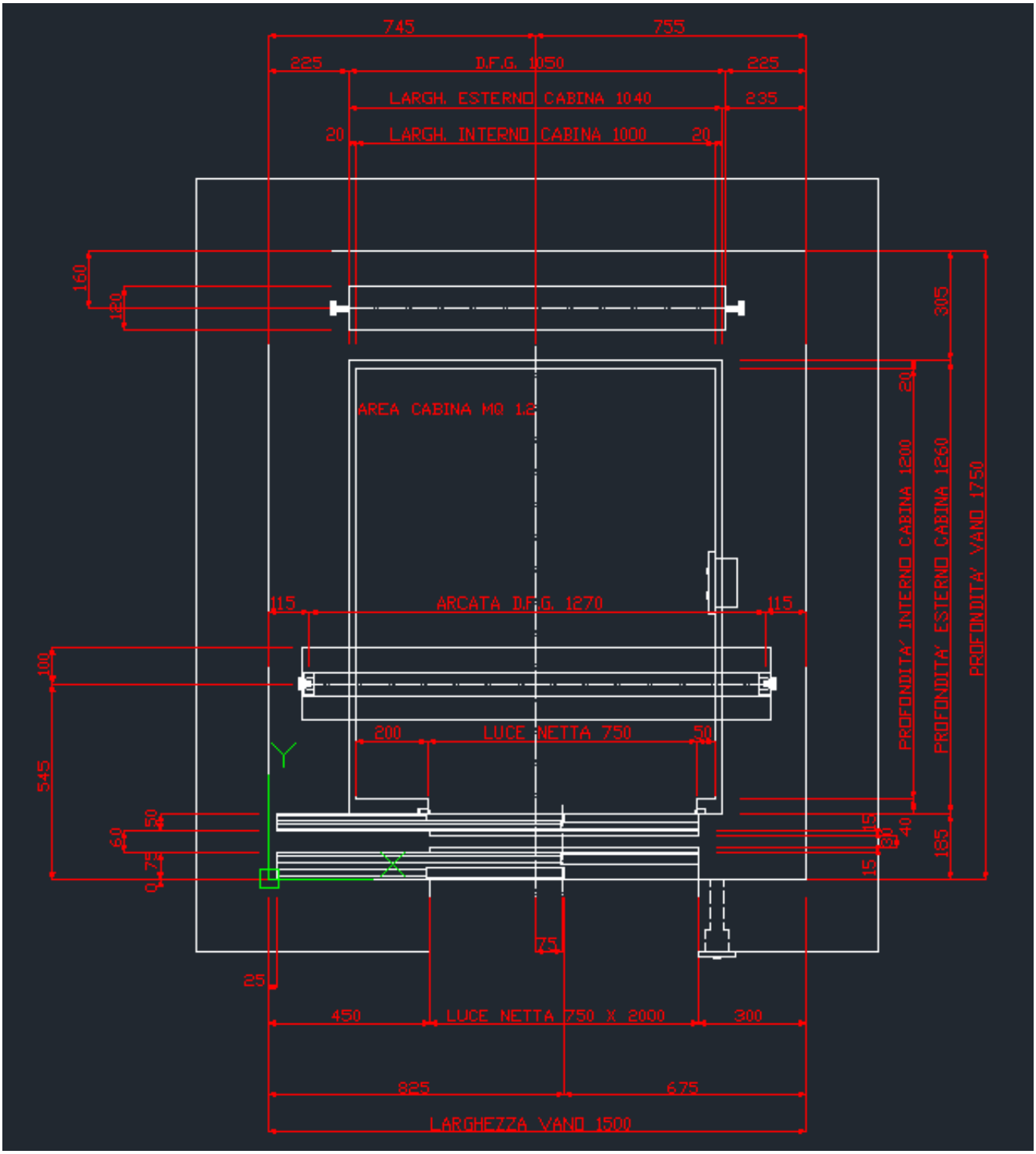

Figura 43

L'ascensore della *Figura 43* è a scopo dimostrativo. Infatti esso riguarda sempre l'impianto preso in esame nelle due figure precedenti, ma una soluzione che non prevede il telaio su ogni piano, quindi, molto utile quando, si desidera un'apertura maggiore rispetto a quella che si ha con il telaio. In questo caso specifico la modifica di togliere il telaio su tutti e due i lati è impraticabile perché la cabina non ha una larghezza sufficiente.

Sempre a scopo dimostrativo in questa figura si possono osservare dei riporti sia sul piano che sulla cabina, al fine di mostrare anche questa frequente pratica.

6. Conclusione

Il tirocinio, ha avuto fin da subito l'obbiettivo di sviluppare una pagina in php, che permettesse l'automatizzazione della creazione di disegni necessari per le relazioni tecniche degli impianti di sollevamento.
Per lo svolgimento della prima parte di questo progetto, è stato necessario studiare a fondo tutti i possibili sviluppi dei diversi impianti di sollevamento, al fine così di capirne meglio il funzionamento e riconoscere quali fossero i vari componenti nelle piantine che poi sono state disegnate in AutoCad.
Nella seconda parte ho approfondito il funzionamento del software di AutoCad, sia disegnando in prima persona, che approfondendo come il software sviluppa i propri file dal punto di vista del codice.
Questo studio mi ha poi permesso, attraverso l'integrazione informatica, di creare il codice dei file di disegno, facendo riferimento ad un sistema di assi cartesiane che tracciava le componenti calcolando le coordinate sulla base delle dimensioni inserite.
Lo sviluppo del disegno, aveva quindi bisogno di una buona conoscenza di tutti i possibili scenari di piantina realizzabili, così da poter creare un modello informatico che prevedesse ogni possibile inserimento di variabile.
Infine con lo sviluppo finale dell'interfaccia utente, ho approfondito la creazione delle pagine web e della relativa connessione ai database.

Da questo studio ho quindi compreso il funzionamento degli impianti di sollevamento e approfondito e applicato le conoscenze informatiche acquisite in vari corsi di: "Basi di Dati" per la creazione del database SQL, "Programmazione a Oggetti" e "Tecniche di Programmazione" per lo sviluppo reale del codice attraverso una struttura ad oggetti che ottimizzasse la programmazione e "Progettazione di servizi web e reti di calcolatori" per la creazione dell'interfaccia d'inserimento dei dati in html e css.
Il corso di "Sistemi di Produzione", mi ha invece dato le basi per la creazione ed interpretazione dei disegni tecnici.

Dati i riscontri ricevuti dalle diverse prove, ritengo di poter considerare il progetto funzionante nel suo scopo e di grande aiuto in termini di tempo per lo sviluppo dei disegni AutoCad.

Bibliografia e Sitografia

SECONDO CAPITOLO

DXF Reference di riferimento per la comprensione delle istruzioni AutoCad:

[1] https://archive.ph/SIWmL

QUARTO CAPITOLO

Sito per le librerie CSS:

[2] https://getbootstrap.com/

Tutte le figure in questa tesi sono state create con il relativo software, parziali di codici dello stesso o specificatamente realizzate al fine di rendere la tesi più approcciabile, attraverso rappresentazioni grafiche.